COMMODORE USERS

Ahoy!

48355 K

$2.50/ CAN.$3.00 JAN. 1985

EXPANDED PROGRAM LISTINGS SECTION!

HAVE A BALL WITH **SPRITE BASICS**
FOUR PROGRAMS

ANIMATION THROUGH **CHARACTER SET FLIPPING** ON YOUR C-64

MIDPRINT YOUR NEW YEAR'S INVITATIONS

UPGRADE EASILY WITH **TAPE/DISK TRANSFER**
FOR THE VIC AND 64

POST HASTE! **ULTRA MAIL**
FOR THE C-64

DON'T BE LATE FOR YOUR DATE WITH **ALICE IN ADVENTURELAND**
ON THE C-64

BACH TO BASICS! **MUSIC TUTOR**
FOR THE C-64

CLAW YOUR WAY **TO THE TOP**
ON YOUR VIC 20

GHOSTBUSTERS
FIRST PUBLISHED PHOTOS!
MAGIC VOICE
PAPER CLIP
AND MORE
REVIEWS

INSERT A **DEVICE # DISCONNECT SWITCH** ON YOUR 1541

THE WORKS! **EXPOSE** OF THE **VIC AND 64 OPERATING SYSTEM**



0 71896 48355 01

You asked for them . . .
CARDCO has them!

LQ/1 & LQ/3 LETTER QUALITY PRINTERS
In response to your demand and need, CARDCO now has available "Commodore-ready "LETTER QUALITY PRINTERS". Just plug them in and print.

Offering standard friction-feed and optional tractor-feed, the CARDCO PRINTERS come complete with built-in interfacing for all Commodore Personal Computers, as well as compatible input for PC, PC jr., TRS-80 and other personal computers.

LQ/1 13" carriage, 15 CPS

LQ/3 11" carriage, 13 CPS

"CARDPRINT" C/?+G PRINTER INTERFACE with Graphics
For printers that are not Commodore-ready, Cardco offers the C/?+G PRINTER INTERFACE including all cables and connectors for the following printers: all new Epson MX, RX and FX series, Star Gemini 10X and Delta 10, Prowriter, C.-Itoh 8510, NEC 8023, Okidata 82, 83, 84, 92, 93 and 94, Mannesman Tally Spirit and MT-160, Seikosha GX-100, BMC BX-80 and the Gorilla Banana. Prints the full Commodore character set; graphics, characters, reversed characters and reversed graphics characters.

"WRITE NOW" WORD PROCESSOR SOFTWARE
An excellent time saver, CARDCO OFFERS THE C02 "Write Now" program with built-in 80 column display. You see exactly what will print. All special codes can be transmitted to printers maintaining justification. Easy full-screen editing; works with any printer.

"MAIL NOW" MAILING LIST SOFTWARE
CARDCO D/01 quickly (in memory) sorts by zip, category, name and state; fully compatible with "WRITE NOW". Other fine features include: User-oriented; menu-driven operation; each disk supports 600 entries. Format can print single, double or triple labels across.

All Cardco Products are available at your local dealers. Write for illustrated literature.



cardco, inc.
300 S. Topeka Wichita, Kansas 67202 (316) 267-6525

"The world's largest manufacturer of Commodore accessories."

Reader Service No. 96

Ahoy!

CONTENTS

DEPARTMENTS

FEATURES

*Includes programs: *Runner* and *Movers* (for the C-64)
**Includes programs: *Sprite Maker, Sprite Saver, Sprite Loader, Sprite Controller* (for the C-64)

PROGRAMS

Cover illustration by James Regan
Illustrations inside screens by James Regan, Kellan Hatch, Kat Reviaska; photo by Liz Benford; Ghostbusters *logo © 1984 Columbia Pictures Industries, Inc.*

Publisher
Michael Schneider

Editor
David Allikas

Managing Editor
Robert J. Sodaro

Senior Editor
Tim Moriarty

Technical Editor
David Barron

Consulting Editors
Ben Bova
Morton Kevelson
Dale Rupert

Art Director
Raoul Tenazas

Associate Art Director
JoAnn Case

Art Production
Christopher Carter
Eve Griffin
Mark Kammerer

Technical Advisor
Bob Lloret

Circulation Director
W. Charles Squires

Advertising Director
Lynne Dominick

Director of Promotion
Joyce K. Fuchs

Controller
Dan Tunick

Managing Director
Richard Stevens

Advertising Representative
JE Publishers Representative
6855 Santa Monica Blvd., Suite 200
Los Angeles, CA 90038
(213)467-2266
Boston (617)437-7628
Dallas (214)660-2253
New York (212)724-7767
Chicago (312)445-2489
Denver (303)595-4331
San Francisco (415)864-3252

AHOY! No. 13, January 1985. Published monthly by Ion International Inc., 45 W. 34th St., Suite 407, New York, NY, 10001. Subscription rate: 12 issues for $19.95, 24 issues for $37.95 (Canada $26.95 and $49.95 respectively). Application to mail at 2nd Class postage rates is pending at New York, NY and additional mailing offices.

Postmaster, send address changes to *Ahoy!*, 45 W. 34th Street, Suite 407, New York, NY 10001. Direct all address changes or matters concerning your subscription to *Ahoy!*, P.O. Box #341, Mt. Morris, IL 61054. All editorial inquiries and software and hardware to be reviewed should be sent to 45 W. 34th St., Suite 407, New York, NY 10001.

VIEW FROM THE BRIDGE

We can hear it now: "'Expanded program listings section'? They've got one more page of programs in this issue than last. That's worth putting in big letters on the cover?"

All right, take a closer look. First, our programs are now listed in 40 columns, not 33 as before, allowing us to squeeze into a single magazine line many program lines that would otherwise have spilled over into two. Then we've eliminated the Scotch Rule that used to adorn the top of our program listings pages, and pushed the type up as far as it can go without the risk of its being trimmed off during printing.

But most important, the familiar stacks of *Bug Repellent* Line Codes that used to follow each program have been absorbed into the program lines (see the white-ruled area to the right of each column). Count the pages occupied by *Bug Repellent* line codes in any previous issue of *Ahoy!* and you'll see how important a space saver that is!

All of which translates into a more expansive lineup of programs, this issue and every issue. But as we increase the quantity of programs offered, we won't decrease the quality. A revolutionary concept for computer magazines, we know, but one we intend to stick to, beginning with our current issue:

• Impossible for a BASIC program to keep track of names, addresses, and phone numbers for labels, cards, and other uses at machine language speed? We thought so, until we saw *Ultra Mail* by Joe Fusco (*Brisk,* July '84) and Bob Lloret (*Ahoy!* programs too numerous to list). (Turn to page 13.)

• Exactly what you'll send to all those people on your new mailing list is up to you, but Bud Black's *Midprint* will enable you to generate attractively formatted cards and announcements on your C-64 or VIC 20. (Turn to page 41.)

• Now we know why we haven't heard from Bob Spirko since our July issue, when he contributed *Checklist.* He fell down a rabbit hole around tnat time, and spent the following six months researching *Alice in Adventureland*—a C-64 graphics adventure that's as cute as a bunny and as challenging as the dickens! (Turn to page 39.)

• Our anguish over how little some would-be *Ahoy!* programmers are willing or able to do with a 64 is assuaged when we see how much certain programmers can do with an unexpanded VIC 20! Kevin Dewey did a great deal in a very short space with four screens, eight levels of play, and a horde of different monsters that block your path *To the Top!* (Turn to page 55.)

• Paul Harker would be equally justified in blowing his own horn over *Music Tutor*, a sound- and graphics-rich program that provides musical instruction on a par with many commercial packages. (Turn to page 37.)

• While Mark Joerger's *Tape/Disk Transfer* for the VIC and 64 is short, the time it will let you save when transferring files from tape to disk is anything but! (Turn to page 25.)

• Our regular columnists have also gotten in on the first monthly *Ahoy!* Program Jamboree by putting the graphics power of the C-64 in your hands (or at your fingertips). The quartet of sprite programs offered by Dale Rupert in the *Sprite Basics* edition of the *Rupert Report* will allow you to create, save, and manipulate these graphic gremlins. (Turn to page 50.) And Orson Scott Card's *Creating Your Own Games on the VIC and 64* discusses *Bringing the Screen to Life* by flipping character sets to create cartoonlike animation. (Turn to page 15.)

This month's roster of feature articles should prove equally mouth-watering, even with the absence of Morton (the K) Kevelson (who needed this month to complete his definitive study of printer interfacing, '85 style—appearing next issue):

We know that those of you who were unskilled at soldering and wiring were so intrigued by Anthony Scarpelli's *Building a Cassette Interface for your C-64* (September '84) that you went out and learned. Good thing, because those skills will once again be necessary for *Inserting a Device # Selector Switch on the 1541 Disk Drive.* Tom Benford tells you how you can shuttle back and forth between two drives at the flick of a toggle switch. (Turn to page 93.) By the way, readers who enjoyed the *Cassette Interface* article will find an intriguing addition in this month's *Flotsam,* by Bob Kovacs, creator of Tymac's The Connection interface. (Turn to page 57.)

Donald H. Graham's *Exposé of the VIC and 64 Operating System* lays bare the works of your computer and clues you in on some helpful programming techniques. (Turn to page 21.)

Ready at last for some fun and games? Robert J. Sodaro previews, way ahead of the pack, Activision's *Ghostbusters* for the 64. (Turn to page 59.)

We have no room to go into detail on our *Reviews, Scuttlebutt, Commodares,* and *S.O.S.* sections. We'll let you explore that much alone.

We almost forgot to wish ourselves a happy birthday. This issue marks one year that we've been coming to you once a month. Don't send any cakes, please. Just cards—and letters—on any subject at all. We love to hear from you.

—David Allikas

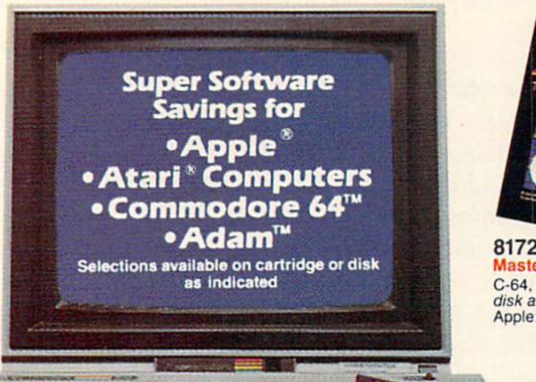
TAKE ANY 2 FOR $4.95 EACH

when you join the Columbia Software Club and agree to buy 4 selections at regular Club prices in the next 2 years



0040022
Addition/Subtraction
0041012
Multiplication/Division
Both available for C-64, Atari H.C. and Apple: *disk*.

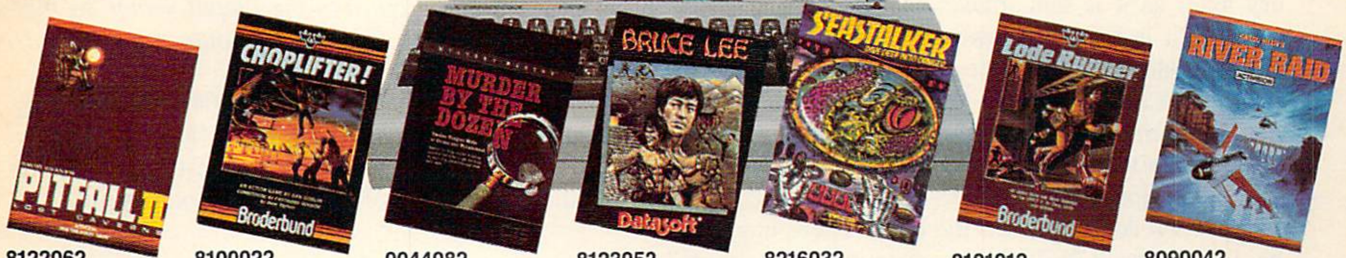
8035022
Zaxxon
Adam, C-64: *cart;* Atari H.C.: *disk and cart;* Apple: *disk*.





8172052
Master Type
C-64, Atari H.C.: *disk and cart;* Apple: *disk*.

8230052
Run For The Money
C-64, Atari H.C. and Apple: *disk*.



8122062
Pitfall II
Adam, Atari H.C.: *cart;* C-64: *disk and cart*.

8100022
Choplifter
C-64, Atari H.C.: *disk and cart;* Apple: *disk*.

0044082
Murder By The Dozen
C-64, Apple: *disk*.

8123052
Bruce Lee
C-64, Atari H.C. and Apple: *disk*.

8216032
Seastalker
C-64, Atari H.C. and Apple: *disk*.

8101012
Lode Runner
C-64, Atari H.C.: *disk and cart;* Apple: *disk*.

8090042
River Raid
C-64: *disk and cart;* Atari H.C., Adam: *cart*.



8105072
Zork II
Atari H.C. and Apple: *disk*.

8104082
Zork I
Atari H.C. and Apple: *disk*.

8150012
Summer Games
C-64, Atari H.C. and Apple: *disk*.

8149052
WarGames
Adam: *cart*.

8102002
Temple of Apshai
C-64, Atari H.C. and Apple: *disk*.

8215042
Beach-Head
C-64, Atari H.C.: *disk*.

8103092
Pitstop
C-64, Atari H.C.: *cart*.

8229082
Pitstop II
C-64, Atari H.C. and Apple: *disk*.

Here's a great selection of sensational software for every member of the family—at super savings! You've seen these software selections in stores anywhere from $19.95 and *up*—yet, now, you can have *any two* for the fantastic low price of only $4.95 each! That's our way of introducing you to the Columbia Software Club—a brand-new service that delivers the best software for Atari® Home Computers, Apple,® Commodore 64,™ and Adam™ systems

How the Club works: about every 6 weeks (up to 9 times a year) you will receive the Columbia Software Club Magazine. In it you'll find out about an exciting variety of the newest software available: simple shoot-em-up adventures, more challenging strategy software, learning software to help the entire family acquire new skills—how to type, master basic math, budget your finances, and much more.

In addition, each issue of the magazine announces the "Pick-Hit Selection"—an outstanding software selection specifically for your system. If you want only this Selection, you need do nothing—it will be sent to you automatically. If you want one of the alternate selections—or nothing at all—just tell us so on the response card always provided, and mail it by the date indicated. You'll always have ten days to make your decision. If you ever receive a selection without having had ten days to decide, you may return it at our expense.

The selections you order will be mailed and billed at regular Club prices—which currently begin at $24.95 and are less than list prices! (A shipping/handling charge and applicable sales tax is added.) Remember, you don't have to buy a selection every time you hear from us—your only membership obligation is to purchase four selections, at regular Club prices, in the coming two years. And you may cancel membership at any time after doing so. If you decide to continue, you'll be eligible for our generous money-saving bonus plan.

10-Day Free Trial: we'll send you complete details of the Club's operation with your introductory selections. If you are not satisfied for any reason whatsoever, just return everything within 10 days for a full refund and you will have no further obligation. So act now!

All applications subject to review, and the Columbia Software Club reserves the right to reject any application or cancel any membership.

COLUMBIA SOFTWARE CLUB, Dept 2RE 102
3000 North 35th Street, Terre Haute, Indiana 47811

Please accept my application under the terms outlined in this advertisement—and send me the 2 selections indicated below, for which I am enclosing check or money order for $9.90 (that's $4.95 for each selection). I agree to buy 4 more selections at regular Club prices in the next 2 years.

Write in numbers of the 2 selections you want ☐ ☐

Send my selections for the following system (check one system only):

☐ ATARI® HOME COMPUTERS 2 ☐ APPLE ® 3
☐ COMMODORE 64™ 4 ☐ ADAM™ 5 (cartridges only)

If you have selected Atari Home Computer or Commodore 64 software—be sure to check the type of format you prefer (check one format only):

☐ CARTRIDGE A ☐ DISK H

Name (please print) ____ First Name ____ Initial ____ Last Name

Address ____ Apt. ____

City/State ____ Zip ____

Do you have a telephone? (check one) ☐ Yes ☐ No

Parent's Signature if under 18 ____

Offer not available: APO, FPO, Alaska, Canada, Hawaii, Puerto Rico

WANT THIS CHARGED TO YOUR CREDIT CARD? Fill in information below, we'll charge the $9.90 and future shipments to your credit card—and credit your account with any returns.

☐ **American Express** ☐ **VISA** ☐ **MasterCard** ____ Interbank No.

Credit Card No. in full ____ Expiration date

Signature ____ **103**

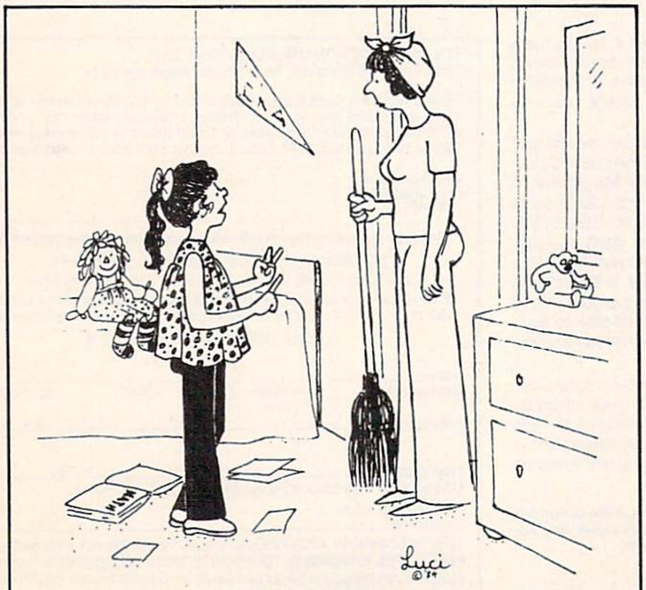
S.O.S.
YOUR PROGRAMMING MAYDAYS ANSWERED

BY DAVID BARRON

HELP! I have a situation where the output of a Commodore 64 is sending data to another device which looks for ASCII code. As everyone knows, the 64 uses "PET ASCII." Unfortunately, the device to which I am sending this information sees all letters as upper case because of the PET ASCII code. It is important that this device be able to distinguish between upper and lower case letters. Therefore, I have written a BASIC subroutine which looks at every letter as it is sent, makes the determination of whether it is upper or lower case from the PET ASCII code, then translates this information into standard ASCII codes for the device that is receiving the information from the Commodore. PROBLEM! This takes an astronomical amount of time and slows down the program considerably as information is read back from the disk. There has got to be a faster way! Will somebody out there help me?

Steven R. Robinson
Chattanooga, TN

The problem you are having is usually solved by using the proper printer interface. In the February '85 Ahoy! *we review several of these devices, which take the "PET ASCII" output of the 64 or VIC and convert it to standard ASCII which is sent out over a centronics parallel port. If you're using a serial printer via an RS232 port, things can get a bit more complicated. In this case you must write a translation routine. BASIC should be sufficient if programmed correctly. Use an array for a lookup table. A$(32), for example, would hold the ASCII equivalent of PET ASCII 32. The most prudent method of calling a routine would be to set a variable such as O$ equal to the string to be printed. Then use a GOSUB to jump to a standard output routine where each character would be looked up and outputted. This should run fast enough to keep up with your printer. If even greater speed is desired, you would have to write a machine language printer driver. Its address would be put into the output vector so that it is called instead of the standard output routine.*



If Jim had 3 bytes, then gave 2 nybbles to Bob and 4 bits to John....

I just purchased a Commodore 64, and was very disappointed. If the machine has 64K of memory, why is only 38911 available for me to use? Where did the other 25K go? Is it hiding somewhere?

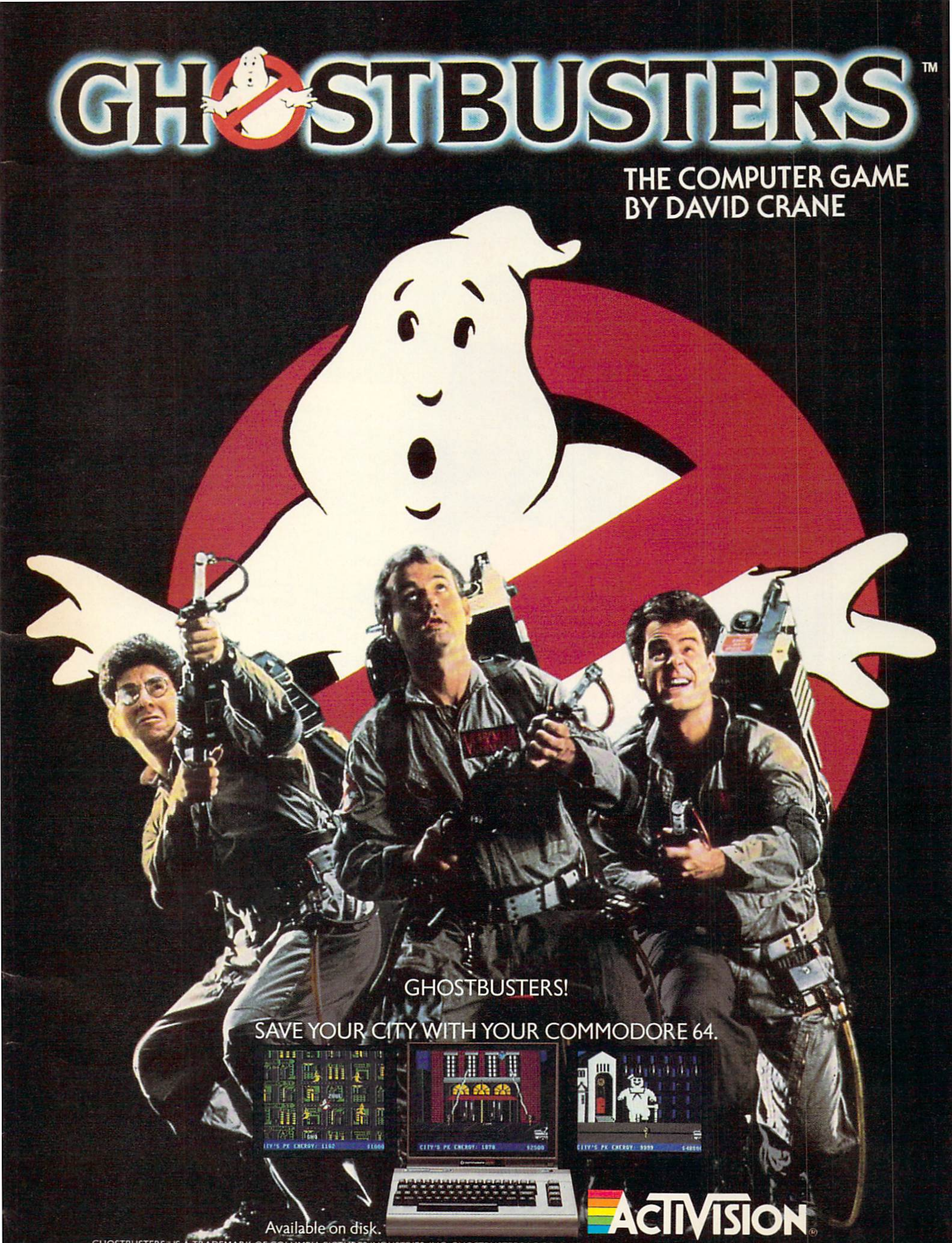
Neil Swift
Tenafly, NJ

You were closer than you realized when you suggested that it might be hiding somewhere, for some of it is. Besides the 38911 available for BASIC, there is a certain amount of RAM reserved for the system's own use (temporary data, pointers, buffers). The screen also uses up a considerable amount of RAM to store characters and character colors. Additionally, there is space for ROM cartridges; and finally, the ROMs that hold the BASIC language. Here's where the fun begins! Behind the ROM for BASIC and the ROM space for cartridges there is RAM. This RAM is not used when programming in BASIC, but from machine language it is possible to switch off the ROM and switch in the RAM, making it accessible. So, Neil, the Commodore 64 really does have 64K of RAM, but not all of it is usable from BASIC.

On my C-64, when I use the collision detect for sprites, sometimes a collision is found when there really isn't one. Why does this happen?

J. Silver
St. Francisville, LA

The problem you're having is related to the fact that you were unfortunate enough to get a "sparkly" 64. Some of the 64's produced generate random bits of garbage on the screen at random intervals called sparkles. If a sprite happens to hit a sparkle then a collision is detected. Unfortunately many 64's have this problem. However there is a way to get around it. Have the program check for a collision twice before allowing a collision to occur. This way you will be sure that your sprite actually collided with an object onscreen and not just a sparkle. (The sparkle will have moved on; the object will still be there.) □



Reader Service No. 85

SCUTTLEBUTT

NEW COMMODORE COMPUTER • TV/MONITOR CHECK • QUICK CASSETTE • PROGRAM RENTAL • DISK NOTCHER • PRINTER-READY CHRISTMAS CARDS • FREE UTILITIES & GAMES • C-64 FAN • MUPPET KEYBOARD • BOOKS • CARRYING CASES • OS CUSTOMIZER • CONTESTS •VIC/64 GAMES

COMMODORE 128

As we send this issue to press, the following item is a scoop. By the time it reaches you, it may still be a scoop, or it may be public knowledge. Rumor-mongering is not an exact science.

The Commodore 128, however, is more than a rumor. It exists, in prototype, at Commodore headquarters in West Chester, Pennsylvania. Don't expect a demonstration if you drop by. The machine is a secret to all except the Commodore inner circle and a handful of privileged outsiders (including our source, always-in-the-know *Ahoy!* writer Tom Benford).

The new computer will feature 128K of available memory, switch-selectable 40/80-column screen, and built-in Z80 (CPM-compatible) co-processor. Unlike the Plus/4, it retains the SID chip. Commodore would like to include built-in software as well, but hasn't found anything good enough yet. Price will be in the $300-$500 range.

Most important, the Commodore 128 will be 100% compatible with C-64 software. A March '85 availability date is the goal; figure that for unadulterated optimism and plan on early summer.

For those who found the SID- and sprite-lacking Plus/4 a confusing progression from the C-64, Commodore's plan at last becomes clear. The Commodore 128 is the true step forward we've been expecting.



Quick Cassette reads data 15 times faster than an audio cassette system.
READER SERVICE NO. 98

HIGH SPEED TAPE DRIVE

Until prices dip well below the $200 mark, disk storage will not make cassette storage obsolete. But Quick Cassette storage might. If Entrepo's under-$85 system lives up to its pre-release publicity, it will read data from tape to computer 15 times faster than an audio cassette system. That would be around 30 seconds for 64K, instead of 23 minutes for an audio cassette—or 2 minutes for the 1541. Data is stored not on conventional cassettes, but on endless-loop "Microwafer" cartridges.

The drive, scheduled for November availability, features an operating system with format, load, save, verify, directory, and clean commands, as well as a built-in utility permitting copying to and from audio cassette, disk, or another Quick Cassette.

Sounds exciting, as long as those Microwafer cartridges don't employ the same technology as the endless-loop tapes many of us are familiar with: 8-tracks. When the lubricant that keeps the tape inside an 8-track cartridge from sticking to itself dries up with age, the tape snaps. Time, and a forthcoming review, will tell.

Entrepo Inc., 1294 Lawrence Station Road, Sunnyvale, CA 94086 (phone: 408-734-3133).

DISK/MEMORY UTILITY

Peek a Byte 64 features a track/sector and computer memory display and editor with cursor control. HELP screens are available with a keystroke. Keyboard errors and disk read/write errors are all reported. Disk; $29.95.

Quantum Software, P.O. Box 12716, Lake Park, FL 33403-0716 (phone: 305-840-0249).



MiraCOOL takes the heat off mother.
READER SERVICE NO. 76

NEW FAN CLUB MEMBER

Last month we told you about fans for your disk drive and other peripherals. But what about your computer? Because the motherboard of your VIC or 64 is sealed in metalized cardboard–eliminating RFI radiation, but cutting it off from the computer's vents–operating temperature can top 120°F.

One source of relief for your memory chips and microprocessor is the MiraCOOL-64 and MiraCOOL-20 fan cartridges from CR Technologies (manufacturer of digital temperature monitoring systems for industry). The appropriate fan plugs into the user port and lowers operating temperature to below 100°F. Price is a cool $52 (add $2.50 shipping/handling for mail orders; Washington state residents add 8.1%).

CR Technologies, 4103 148th SW, Lynnwood, WA 98036.

TALK TO ME

Joining the rising chorus of speech synthesizers for the 64 is The Voice Messenger, a $49.95 unit boasting two voice selections, built-in software that lets your 64 talk as soon as you turn it on, and an infinite vocabulary. This last is made possible by an allophone-based synthesizer chip that lets individual speech sounds be strung together to form any English word.

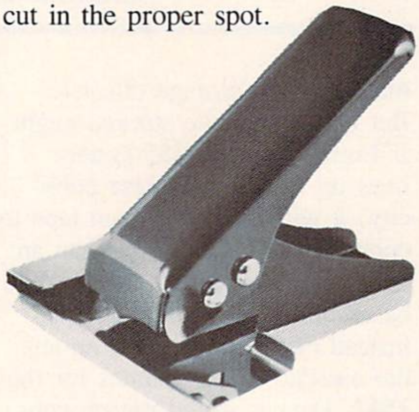
Currah Technology, 50 Milk Street/15th floor, Boston, MA 02109 (phone: 617-357-8084 or 800-443-0100/ext. 797).



Voice Messenger plugs in cart port.
READER SERVICE NO. 77

DOUBLE YOUR DENSITY

The Disk Notcher will cut a write enable notch on the left edge of your disk where the manufacturer doesn't cut one, enabling you to use the side of your disk that the manufacturer doesn't want you to use. Of course, you could do this by hand and save $29.95. But the Notcher insures an even cut in the proper spot.



The Disk Notcher cuts one for you.
READER SERVICE NO. 78

Quorum International, Industrial Park Station, P.O. Box 2134, Oakland, CA 94614 (phone: 1-800-821-5226/ext. 404).

NO FILE LIKE HOME

Didn't we say it exactly one year ago, in the January '84 *Scuttlebutt* (bottom of page 98)? "Graphics programs obviously have virtually endless possibilities. What about a program for interior decor–combining furniture, wallpaper, and carpets of different styles and colors? Or an architectural program for designing houses and skyscrapers?" Joyce Hakansson Associates and CBS Software must have been listening. Our dream program and their *Dream House* are one and the same.

The user builds and decorates his or her fantasy home, starting with a blueprint and ending with a decorated, landscaped domicile that can range from a Manhattan penthouse to a San Francisco Victorian. Included are animation and sound effects, paint system, icon-driven design menu and a 28-page Designer's Guide. Price is $39.95.

CBS Software, One Fawcett Place, Greenwich, CT 06836 (phone: 203-622-2500).

THE NEW TECH TIMERS

Not even the timer you use to turn appliances on and off has been left untouched by the age of technology. Savergy's Powerport will allow your VIC or 64 to do the regulating from now on. The under-$100 system plugs into the user port and accepts eight AC or DC loads. Included is *Autoports* time-of-day scheduling software for either the 64 or VIC, on either disk or cassette.

Savergy, Inc., 1404 Webster Ave., Fort Collins, CO 80524 (phone: 303-221-4200).

TEST YOUR TV/MONITOR

Honorable mention for the list of Great Lies: "It's gotta go back to the shop." But the next time your TV repairman pronounces that diagnosis, you can get a second opinion with *Video Tune-Up*. The tape- or disk-based program

turns your C-64 into a video analyzer for aligning, setting up, testing, or servicing your TV or monitor. It generates the standard video test patterns (crosshatch, purity field, color bars, etc.) and provides an animated video performance test. Included for the $39.95 price is *Bar Sweep 64,* which generates a hi-res test pattern for checking the video response of a monitor/TV.

Solas Products, P.O. Box 665, Spring House, PA 19477 (phone: 215-362-5028).

CHECK IT OUT

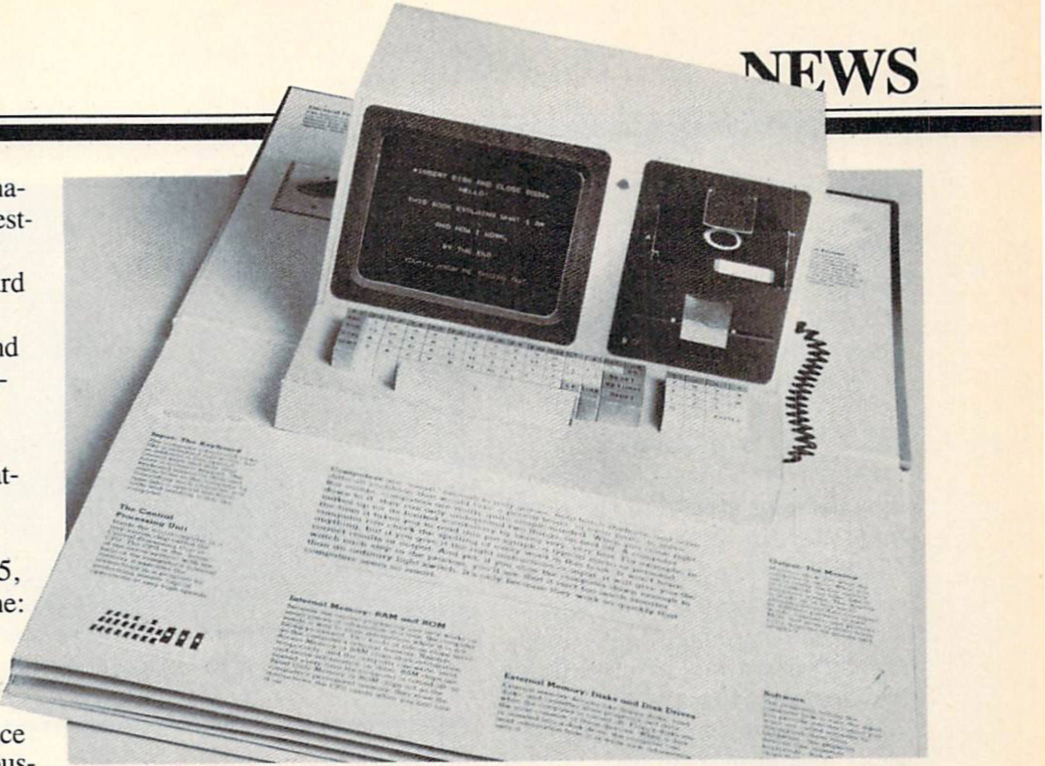
Videocassette rental was once a novel idea. Now it's bigger business than videocassettes sales. Will the same happen with floppy disk rental? Unlikely. Computer programs aren't worth much on one-night stands. But companies like Educational Computing Network are testing more appropriate rental schemes.

Their *Commodore 64 Rental Collection* consists of 24 disks comprising over 200 educational programs and games, all public domain, all copyable. Fee is $49.00 for 30 days. Further information and a free catalog can be had by sending a self-addressed, stamped envelope to Educational Computing Network, P.O. Box 8236, Riverside, CA 92515.

HARD COPY

Why limit your computer reading to *Ahoy!*? With books like the recent releases described here, you have no excuse for not further expanding your Commodore I.Q.

PcDex: Magazine Resource Guide for Commodore 64, VIC 20, and PET/CBM Personal Computers indexes 12 Commodore-related magazines published between January '82 and April '84. $14.95 from Altacom, Inc., P.O. Box 19070, Alexandria, VA 22314 (phone: 703-683-1442).



One of **Inside the Personal Computer***'s six entertaining "pop-up" spreads.*
READER SERVICE NO. 79

Computer Programming for the Compleat Idiot provides the beginner with step-by-step instructions, programs demonstrating standard tasks, and guidelines for creating an original program. $10.95 from Design Enterprises of San Francisco, P.O. Box 14695, San Francisco, CA 94114 (phone: 415-282-8813).

Expanding Your Commodore 64 Capabilities details expansion options in the areas of telecommunications, mass storage, monitors, word processing, and database management. $11.95 from Prentice-Hall, Englewood Cliffs, NJ 07632 (phone: 201-592-2640).

Big Computer Games lists 12 games adaptable to most microcomputers. $9.95 from Creative Computing Press, One Park Avenue, New York, NY 10016.

The Second Self: Computers and the Human Spirit explores the social and psychological implications of the computer culture. $17.95 from Simon and Schuster, 1230 Ave. of the Americas, New York, NY 10020 (phone: 212-245-6400).

Let's Learn BASIC: A Kids' Introduction to BASIC Programming (C-64 version) uses the challenge-and-reward approach to motivate 8-14 year olds to learn. $8.95 from Little Brown and Company, 34 Beacon St., Boston, MA 02106 (phone: 617-227-0730).

Assembly Language for Kids: Commodore 64, for junior and senior high schoolers, includes its major programs on a disk. $14.95 from Microcomscribe, 8982 Stimson Court, San Diego, CA 92129 (phone: 619-484-3884).

VIC 20: Easy Guide to Home Applications explains and lists 48 programs, 4 fun patterns, 10 educational, and 3 household programs. $8.95 and $11.95 respectively from Howard W. Sams & Co., Inc., 4300 W. 62nd St., Indianapolis, IN 46268 (phone: 317-298-5400).

Inside the Personal Computer consists of six "pop-up" spreads (among the most elaborate you will ever see), illustrations,

charts, and diagrams that explain the functions of the computer. $19.95 plus $2.50 shipping from Sweet Gum, Inc., 15490 N.W. 7th Ave., Miami, FL 33169 (phone: in FL 305-687-9338; rest of USA 305-687-9338).

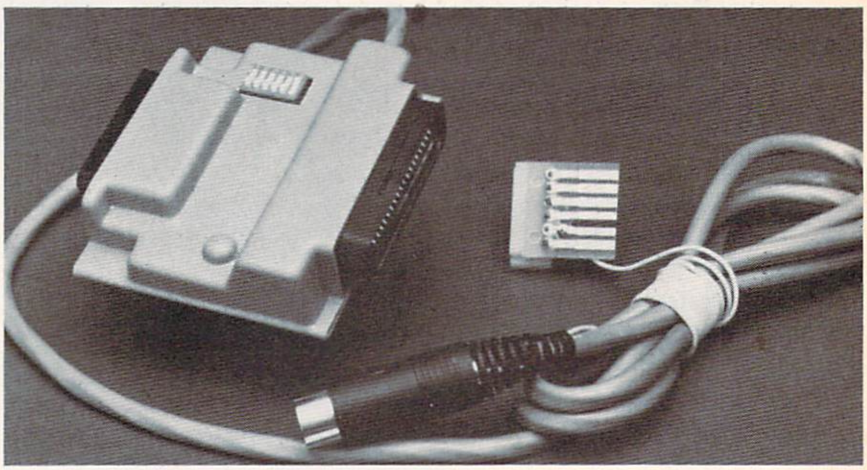
OS CUSTOMIZER

Once you've learned all about the operating system of the C-64 (and VIC 20) by studying Donald Graham's article in this issue, you can reconfigure the entire thing with *The Operating System* from Hacker's Hardware. The cartridge-based customizing component adds escape keys, quick load keys, default to disk drive, automatic machine language loading with display of load addresses, and improved video displays. Other features can also be accessed. Over 18 ML-speed utilities are available, including partitioning of the C-64 memory into separate blocks (enabling the user to access two BASIC programs at once) and automatic creation of loader programs that can run both BASIC and ML programs. Price is $54.95, or $64.95 for version with customized text display.

Hacker's Hardware, P.O. Box 7933, San Diego, CA 92107.

UNIVERSAL INTERFACE

Cardco's new Card/?PS universal parallel/serial printer interface will work with any standard serial input printer that can accept data at 300/1200 baud, and it will convert CBM ASCII to standard ASCII if desired. But at least as important as what it will do, is what it won't do. It won't tie up your modem port or expansion port, because it plugs into the serial bus port on your 64, VIC, or disk drive. And it won't require driver software, because it uses standard Commodore print format. Retail price is $79.95.



Card/?PS universal parall/serial interface uses Commodore print format.
READER SERVICE NO. 80

Cardco, Inc., 300 S. Topeka, Wichita, KS 67202 (phone: 316-267-6525).

PC ON EARTH

The no-frills greeting cards generated by Bud Black's *Midprint* (see page 41), while fine for Veronica's Sweet Sixteen, are a little tacky for Christmas. But if you're determined to send holiday greetings that are both chic and personalized, whatever the cost, Compugreet's line of printer-ready Christmas cards should work well with *Midprint* or with your word processor and some imagination. The cards (which are also available for birthdays and other occasions) feature an illustrated front, a blank inside, and tractor holes on the sides. 20 cards with envelopes for $9.95, 100 for $45, 300 for $130.

Compugreet, P.O. Box 3357, Reston, VA 22090.

HOST OF GAME SHOWS

Three educational releases for the C-64 from Dynacomp:

Game Show USA–Presidents questions one or two players on various chief executives' years in office, nicknames, VP's, and more. Both it and *Game Show USA–States* award points based on correctness and speed. Each $14.95 cassette, $18.95 disk.

The Science Professor, for children up to grade 9, comprises 10 programs on insects, machines, bones and muscles, matter, rocks, atmosphere, and more. Each program requires 16K. Diskette is $99.00.

A STEAL

Computer Ed. is offering free word processing and business graphics programs with the $64.95 purchase of their *Hot Accounts* home/small business financial management system (see January '84 *Scuttlebutt*).

The word processor includes over 60 commands, screen editing, rapid scroll, search, and a preview option allowing the user to see the text as it will print. The business graphics program makes possible screen presentation of 3D displays, data filing, and chart storage. Both free programs will be sent by mail to users who return their *Hot Accounts* warranty registration.

Computer Ed., 1002 Brookes Avenue West, San Diego, CA 92103 (phone: 619-295-4145).

Continued on page 32

ULTRA-MAIL

FOR THE 64

23¢ USA

BY JOE FUSCO AND BOB LLORET

INTRODUCTION BY BOB LLORET

Did you ever desire to accomplish something that others said couldn't be done? Well, it's a known fact that BASIC is a much slower programming language than machine (or if you prefer, assembly) language. *Or is it!*

By now, ML programmers are probably ready to hang us, if only in effigy. Relax, fellows, we are in complete agreement with you. However, armed with a vast knowledge of storing and retrieving information and quite a bit of talent, Mr. Fusco bravely embarked on his task. Believe it or not, after months of experimentation and fine tuning, Joe's labors were rewarded with a program which, when matched to a best-selling database manager, retrieved stored information faster than the ML program. How was this accomplished? Surely not by the speed of BASIC, but rather by the keen programming techniques that Joe used to maximum effect in the creation of *Ultra-Mail.*

While speed was a prime factor, Joe (with my assistance) tried to give attention to all the details of a well-written, user-friendly program. What we accomplished, we're both very proud of. Any C-64 user who wants to keep track of names, addresses, and phone numbers for labels, cards, an address or phone book, a customer list, or any similar business application would be well-advised to put this speedy and simple-to-operate program to work!

As far as instructions go, the program was specifically designed to be well-prompted, and easy to follow the first time through. However, I'll provide some explanation of some of its functions.

When you use *Ultra-Mail* for the first time, it will be necessary for you to create a data disk in order to store the information. *Do not use the program disk for storage.* Insert the program disk into drive #8 and type LOAD "*",8. The drive will search for the first program on the disk and load it into the computer. When the ready prompt comes on the screen, type "RUN", and the message INSERT DATA DISK IN DRIVE #8 will appear. *Remember, all information previously stored on this disk will be erased.* After inserting your data disk, press "D" and the screen will display the message "FORMATTING DISK". After formatting the disk, the program will create the files on the disk it needs to store the information. The message "CREATING DATA FILES" will be displayed along with the numbers 0 and 50. If your screen shows any numbers other than 0 and 50 you have a problem with your disk or a hardware problem inside the drive. Once this is done the message "INSERT PROGRAM DISK IN DRIVE #8" appears. At this point type LOAD "MAIL",8 and then type "RUN". *Please note!* The *only* time you use this procedure is when you want a *new mailing list* on a *new data disk*. At all other times load the *Mail* program and run it. It will display the title screen. When you're ready, press F1 and you'll go to the main menu. From here you can go anywhere in the program. The selections do as their names imply and are well-prompted: 1—*Add*—add a name. 2—*Change/Inquire*—change information on a name or check on information. 3—*Print Commands*—to print one name, one label, or the whole file in or out of alphabetical order. 4—*Delete*—to delete any name. 5—*Search/Scroll*—to search and list the whole file onscreen. 6—*Sort*—to sort the file. 7—*Exit program*—*must* be used whenever you are finished using the program to update your file.

SEE PROGRAM LISTING ON PAGE 64



SMART START™. . . REALIZE THE GREATEST POTENTIAL OF YOUR COMMODORE 64

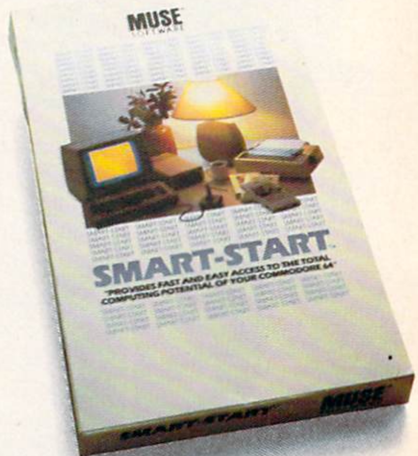
Sound, graphics and other advanced features are what make the Commodore 64 as popular as it is. But these features can also require a lot more programming know-how than many Commodore owners care to learn. That's why MUSE® developed SMART START™—to harness the power of the Commodore and put it conveniently at your finger-tips.

Here are just a few of the things you can do with SMART START™:

- You don't need to be familiar with the Commodore keyboard, SMART START™ lets you quickly select all functions from pull-down menus with cursor or joystick.
- No need to type long instructions to LOAD, RUN, SAVE and CATALOG programs, with SMART START™ you press a single key.
- Easily design colorful sprite graphics and exciting sound effects. SMART START™ automatically generates the BASIC code for use in your own programs.
- Keep track of time with an on-screen digital clock and alarm.
- Easily create exciting and unique sound effects.
- Merge multiple programs into one.
- Easily print out screens and programs.

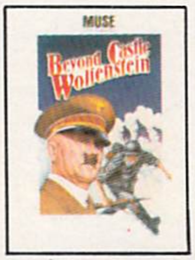
Get the most out of your Commodore 64 with a SMART START™ from MUSE.®

(Joystick and printer with parallel interface optional)











Super-Tex® Professional and Castle Wolfenstein™ are available for the Apple II+, IIe, IIc; Commodore 64; Atari 400/800/1200XL; IBM—PC, PC jr. Beyond Castle Wolfenstein™ is available for the Apple II+, IIe, IIc; Commodore 64; Atari 400/800/1200XL. SMART START™ and Space Taxi™ are available for the Commodore 64.

Apple, Atari, Commodore and PC and PC jr. are registered trademarks of Apple Computer Corporation, Atari Inc, Commodore Business Machines and IBM respectively.

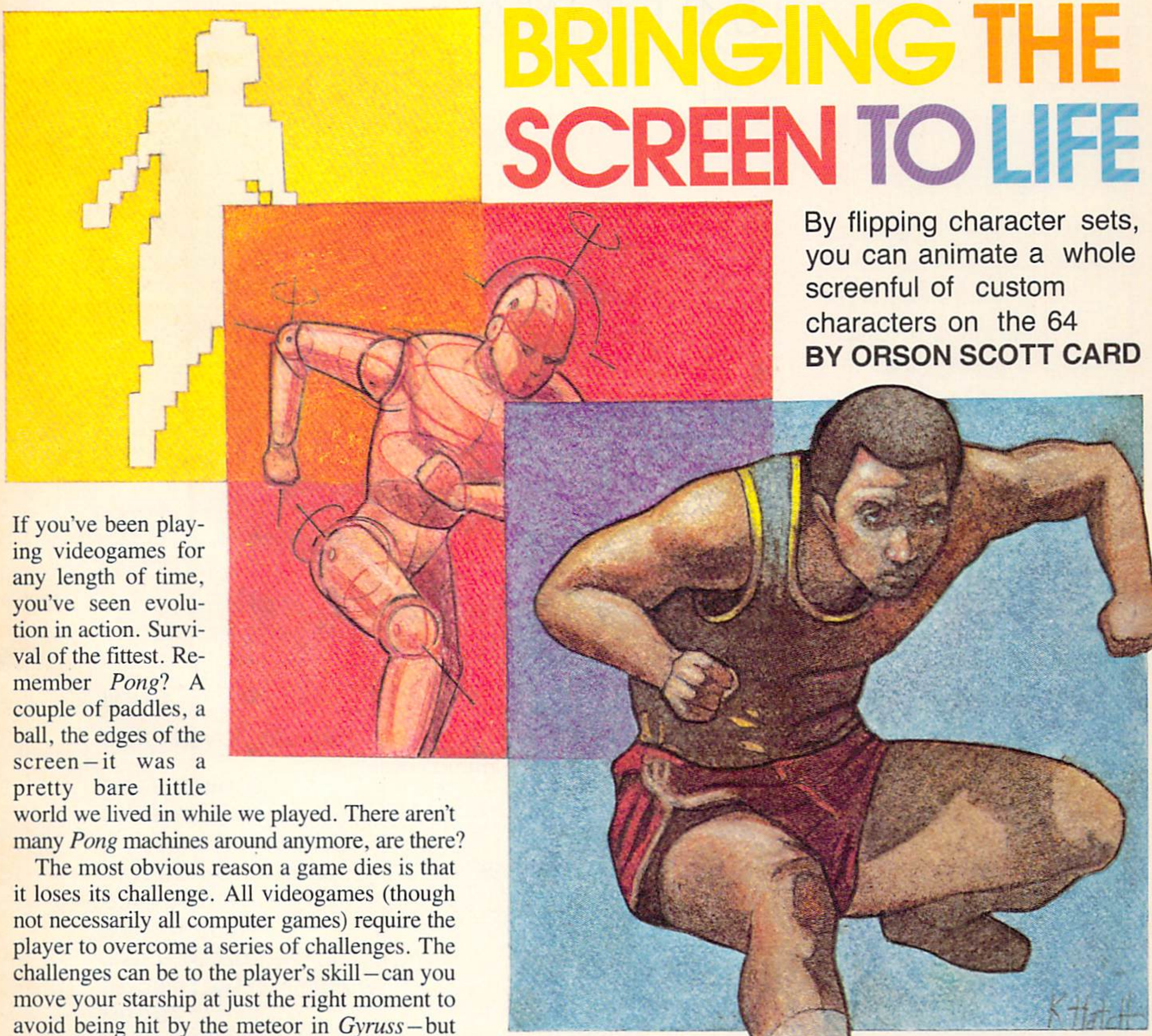
EXPECT THE EXCEPTIONAL...

MUSE® SOFTWARE

347 N. Charles Street
Baltimore, MD 21201
(301) 659-7212

Reader Service No. 102

BRINGING THE SCREEN TO LIFE



© Kellen Hatch 1985

By flipping character sets, you can animate a whole screenful of custom characters on the 64

BY ORSON SCOTT CARD

If you've been playing videogames for any length of time, you've seen evolution in action. Survival of the fittest. Remember *Pong*? A couple of paddles, a ball, the edges of the screen—it was a pretty bare little world we lived in while we played. There aren't many *Pong* machines around anymore, are there?

The most obvious reason a game dies is that it loses its challenge. All videogames (though not necessarily all computer games) require the player to overcome a series of challenges. The challenges can be to the player's skill—can you move your starship at just the right moment to avoid being hit by the meteor in *Gyruss*—but leave it there long enough to hit enough enemy ships? The challenge can also be to the player's mind—can you figure out how to get through a maze despite the obstacles? Especially in the arcades, videogames seem to work like puzzles—you play a new game over and over until you "solve" it. While the players are still working on solutions, the game makes money; when the puzzle is resolved, the income falls off and very quickly the game is replaced with another that's still hot. When is a game "solved?" After all, most games will go on to infinity, if you can train yourself to play that well. In theory, you could play *Joust* forever—though I have a way of bumping into the pterodactyl's mouth too often to hope for immortality. Since the game will go on forever, getting ever faster and more challenging, when have you beaten it?

I believe a game loses its challenge, not when we can play eight days straight without ever losing, but rather when there is nothing left to discover about the world of the game. I just got through a grueling three-day bout with *Boulder-Dash.* It's a wonderful game in which a little cartoon character named Rockford must pick up a certain number of diamonds on each game level before the time runs out. To complicate things, the screen is well-supplied with boulders, which can fall on Rockford and mash him. Also, there are fireflies and butterflies wandering around, which blow up if Rockford gets too close to them or if boulders fall on them. When butterflies blow up, they turn into diamonds—and on some levels, that's the only way to get any diamonds at all.

The game has twenty different screens on each of five levels. On each level the screens come in the same order—but

Phoenix Red
SOFTWARE



ADVENTURE GAME TUTORIALS

Disk I

TEXT ADVENTURES

- ZORK I
- ZORK II
- ZORK III
- ENCHANTER
- SORCEROR
- STARCROSS
- SUSPENDED
- PLANETFALL
- DEADLINE
- WITNESS

Only $9.99!

Disk II

MEGAVENTURES

- ULTIMA II
- ULTIMA III
- ULTIMA IV
- RETURN OF HERACLES
- QUESTRON

Only $9.99!

Disk III

GRAPHIC ADVENTURES

- GOTHMOGS LAIR
- MASK OF THE SUN
- DEATH IN THE CARIBBEAN
- BLADE OF BLACKPOOLE
- CRITICAL MASS
- GRUDS IN SPACE
- ULYSSES AND THE GOLDEN FLEECE
- WIZARD AND THE PRINCESS
- ASYLUM
- THE INSTITUTE

Only $9.99!

Have you ever played an Adventure Game where you just plain got stuck and couldn't get any further? Well, here's the answer — Phoenix Red's Adventure Game Tutorials. They can be used as hints or they will tell you exactly what to do to get out of all those sticky situations. These Tutorials are written for **your** computer (A Commodore 64) not someone elses, so they **work.**

We pay postage and handling.
A must-have for all adventures.
Order yours today!

"We put out money where our software is, not our copy protection."

— Phoenix Red

Send check or Money Order to:

Phoenix Red Software
2705 Oak Dr.
Houston, Texas 77539

Call 713-280-8113 for more info.

Reader Service No. 75

with major changes in the layout, the timing, and the number of diamonds to be acquired, so that each level is like a whole new game. I had fought my way through levels 1 through 4. Now I was battling through level 5, mastering each new screen as I came to it. A couple had stymied for for hours. But I had got to the next to last screen, mastered that in a day or so, and now was fighting my way to a solution of the last screen in level 5.

And then I got it. On earlier levels, I had had to blast holes in the walls of two boxes, clear them out, and then drop boulders through a magic wall to make diamonds. Now I realized that on this level, I only had time to clear out and drop boulders through the righthand box. Once I understood that, it was almost easy. I had finished the last screen of the highest level. I was finished. My wife said, "Does this mean you can talk to the children again?" My marriage was saved, my family was saved—I no longer had to play *Boulder-Dash*.

Deep in my heart, though, I was hoping for more. I had loved playing the game, not just to solve the puzzles, but because I liked the way its world worked. Even levels and screens I had long since mastered were still fun, because I liked doing the things that Rockford had to do—leading butterflies into traps that turned them into diamonds appeals to the Br'er Fox in me, I suppose. And I was hoping that when I completed level 5, the highest level you can select from the opening menu, there would be a level 6.

Instead, level 5 simply repeated itself again. I played through it three times, hoping for some variation. But that was it. That was the end. I had seen all that the world of *Boulder-Dash* had to offer. Then, and only then, was the game over for me.

Not when I knew I could beat it, you see, but when there was nothing left to discover.

The same thing happened to me with *Donkey Kong*. Once I got past the last level that introduces a new screen, I realized that the game was just repeating the same old screens—there would be nothing new. There were still challenges to overcome, but I wasn't interested anymore. I had seen all the gamewright had to show me.

Those who play for points or for their initials on vanity boards (and I do like to see little OSCs on arcade games at the local 7-Eleven) may not understand this. But once I saw another player, a real wizard, take *Galaga* to the point where it repeated. I stopped *Galaga*, even though I had never got above 150,000 points myself. The challenge was still there—but there was nothing new to see, and so I didn't care anymore.

So it isn't just the appeal of a puzzle that draws players into a game and keeps them playing. We will often play on after the puzzle is solved, if there are new things in the world, or if the world is especially appealing; we will often quit before the puzzle is solved, if the world is dull and empty.

SO NOW WE'RE ALL ARTISTS

This is why gamewrights are not programmers, or not *merely* programmers. There is an art to pure programming—what purists call "non-trivial" programs, like word processors and databases and spreadsheets. You can do things cleanly and deftly; you can solve problems in new and insightful and, yes, creative ways.

But clean and clever programming is not enough to make you a good gamewright. Why, it's not even the beginning. Because once a good programmer has seen a videogame, it's easy enough to duplicate it. *Any* competent programmer could create, say, *Donkey Kong*—once he or she had seen and played it.

Thinking of it, though—that's another matter. Creating a full-fledged, believable, attractive world—that takes the same kind of creativity required to direct a movie or make the sets and special effects for a fantasy world that had never been seen before. Think of the book and movie *Dune*. The writing in that book isn't great. Even the plot is a little threadbare. But the world—ah, the world was rich and full, dangerous and beautiful, and as I read the book I never wanted to leave. I dreaded seeing the movie, for fear that the filmmakers would make a botch of it. That they would be shallow or glitzy. Because that world was a good place to live, and I didn't want someone to wreck it for me.

Every creative artist creates worlds. Painters and sculptors do it; even composers and musicians create a world of sound for us to dwell in for a time. And it is in the area of world creation that game design steps over the boundary from programming into art.

Because the same kind of excitement that gripped me when, at the age of fifteen, my older brother got me to read *Dune* gripped me also last Christmastime, when I stopped at a convenience store on the way home from a meeting in Danville, Virginia. I saw a three-dimensional castle and a stupid-looking bear running around, dodging witches, mobile trees, swarms of bees, and other bizarre dangers. It was, far and away, the most complete world I had ever seen in a videogame.

I played for half an hour, vowing to find the game back home in Greensboro and play again. I couldn't find it. And on my next trip to Danville, I couldn't even find the convenience store. Or rather,

I found it—but in less than a week it had been boarded up. I asked some other people, and they told me the store had been closed down for a long time. I began to think of it as a *Twilight Zone* experience. The perfect videogame, discovered in a place where it couldn't be—

But then reality set in. I found *Crystal Castle* in an arcade in Orem, Utah, and I realized—to my relief—that I had not hallucinated it.

Why, though, had it haunted me in much the way that a good book or movie haunts me? It was just another "cute" videogame, to use the disparaging industry term for fully animated games. Bentley Bear was just *Donkey Kong, Jr.* all over again, wasn't he?

No, he wasn't. *Crystal Castle* was a superior game because the animation was superb; the illusion of three dimensions was nearly perfect; and above all, there were many, many different castles and I was free to explore them virtually at will. Gone were the confining walls of *Pac-Man.* Gone were the set patterns of *Donkey Kong* and his kid. There were dozens of ways to conquer each castle, and dozens of things to do. I liked that world, it appealed to me, I liked living there for a long time.

Now, *Crystal Castle* may not appeal to you the same way. My personal quirks are involved—I like moving trees, for instance, and castles appeal to me. The world that captures *you* won't be the same as the world that captures *me.* Even if we like the same game, it'll be for different reasons. But that's fine—that's the way books and paintings and music and movies work, too. Lots of people may like them, but no two of them for the same reason.

That's what you're doing, though, when you create a game. Creating a world for other people to live in. And for that world to work, you need to make it fascinating, believable, and a pleasure to explore.

ANIMATION

You can't make your game as realistic as movies can, of course. You can't point it at something and shoot a picture. The makers of movie special effects only have to make one realistic model and then shoot it for however long it takes to do the scene. But, like film animators, on the computer you have to draw *each frame* of your pictures. If you want to show a man walking, you have to draw him once

standing still, then draw his position a moment later, one leg slightly moved, and then again, with the leg moved more, and so on, until the whole sequence is complete.

You have the same limitation that animated filmmakers have—every detail has a price. Instead of money, though, you pay in memory and speed. Still, we can use some of the same compromises that Hanna Barbera, Warner Brothers, and Disney have used:

1) If an action is repeated over and over, use the same pictures over and over. That is, once you have a sequence of a figure walking, you can use the same sequence twenty times to make him walk twenty steps.

2) If most of the picture doesn't change, you only have to redraw the parts of it that *do* change. Animators use cels on a background; the cels can be moved around and interchanged to create each new image without having to draw more than a few parts of the picture each time.

Redrawing the picture every time would make animation far too expensive ever to do. The same thing with computer animation: you cannot animate the full screen by redrawing it frame by frame. There isn't enough memory in your computer or on your disk to animate more than a few seconds' worth of action if you have to redraw the whole screen.

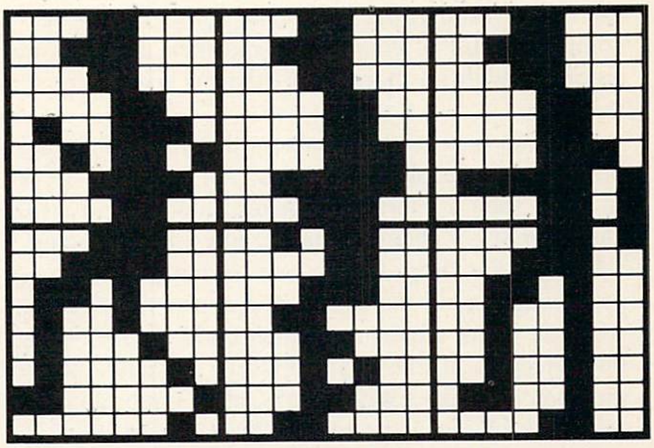
So instead, we change only *parts* of the screen. We've already worked with custom characters, which let us create a few "building blocks" that we can use over and over again by PRINTing our new characters to the screen many times. We've already moved the custom characters around on the screen.

Now comes a programming technique that with a single POKE lets you change every single character in the character set at once. By doing it several times in a row, you can animate an entire screen at once, bringing it to life without noticeably slowing down your program.

It does eat up memory, and you can't have more than seven stages in the complete animated cycle. But as you'll see with the two demonstration programs on page 72, it's enough to make some really dazzling effects without elaborate programming.

CHARACTER SET FLIPPING

Last month we created custom character sets, and told the computer where to find the new character set by POKEing its location code into location 53272. Now we'll create several character sets, and put them in order in memory. If we had three character sets in our sequence, we would put one at lo-



cation code 10, one at 12, and one at 14.

Then, within each set, the same character would stand for a different stage in the animated sequence. For instance, let's say we are creating a four-stage bouncing ball. Stage 1 has the ball high in the air. Stage 2 has it halfway to the floor. Stage 3 has it on the floor, partly flattened. Stage 4 has it back up again, halfway to the top.

So in the first character set, the letter A would be replaced by the pattern for stage 1. In the second character set, the letter A would have the pattern for stage 2, and so on, through the four character sets.

Your program PRINTs the letter A on the screen wherever you want the bouncing ball. You can have one bouncing ball, or dozens. They will *all* bounce at once when you cycle through the character sets. You just POKE each character set's location code into 53272 in turn, and the entire screen changes at once—without your having to PRINT anything different on the screen. One POKE animates the entire screen!

The first demonstration program, called *Runner,* is a three-stage animation of a figure running. The left-facing version of the runner is shown in the figure. Notice that the figure consists of two characters, one atop the other. The only motion is in the legs and arms—the body stays in the same place. The right-facing characters are identical, except they are the mirror image.

The second demonstration program, called *Movers,* shows a four-stage animation of seven different figures. There are:

A bouncing ball
A spinning bar
A spider moving up and down
A cone that rises and falls
A circle that pulsates
A skull that opens and closes its mouth
A snake that goes around in circles
All of these are animated with a single POKE

each time through the loop. The screen could be completely full of characters, and they could all be animated with no more effort than it takes to animate one.

You can also have characters that *aren't* animated on the screen at the same time. All you have to do is make such a character's pattern identical in all the character sets. Then no matter how often the character sets are flipped from one to another, that character will remain the same. These programs do that with the blank character, which consists of nothing but zeroes in all the character sets. Because most of the screen is filled with blanks, it is important that the blank character be blank—in every set.

MOVING THE VIDEO BLOCK

In both programs, the actual addresses of the character sets are stored in the variable array CB*(n)*, and the location code is stored in array CM*(n)*. The location of screen memory is held in variable SB; the variable SK holds the code that must be POKEd into location 53272 to tell it where screen memory is. By adding SK to each value of CM*(n)* in line 445, we turn those values into the combined screen and character set location code. Then, in the main loop from 100 to 190, we need only POKE 53272, CM*(n)* to change the character set.

Unfortunately, things can't stay that simple. That's because character sets eat up memory—2K at a time. Four character sets use up 8K of memory, and if we put them where we put the custom character sets last month, in the highest space available up to the 16K boundary, we'd end up with less than 6K for our program. That isn't enough space to do much of a game.

So we have to pick up video memory, lock, stock, and barrel, and move it somewhere else. Which means we have to move the Video Control Block.

The VIC-II chip, which controls video in the 64, can only look at 16K of memory at a time. Within that 16K it expects to find the character set, screen memory, color memory, the sprite patterns, and the bit-map (if there is one).

The 64K of the 64's memory includes four possible video blocks:

Block	Starting address	Code
0	0	3
1	16384	2
2	32768	1
3	49152	0

Block 0 is the default video block. The screen memory and character memory codes consist of the number of the 1K boundary where that region of memory starts within the block. Then that number is POKEd into 53272. For instance, if we put screen memory at the 15K boundary and the character set at the 12K boundary, we would POKE 15*16 + 12 into location 53272. (The screen location code is multiplied by 16 before being added to the character memory code. Remember that character memory must be at an even-numbered boundary: 0, 2, 4, 6, 8, 10, 12, or 14.)

Now, if the video block is block 0, the VIC-II will look for screen memory at 15360 (15*1024) and character memory at 12288 (12*1024). But if the video block is block 1, then the VIC-II will look for screen memory at 15*1024 + 16384, or 31744. In other words, the code you POKE into 53272 represents which 1K boundary *within the video block* contains screen and character memory.

(The operating system fools the VIC-II into thinking that color memory and the ROM character set are within the video block, even though they're not.)

Because blocks 2 and 3 are full of ROM, there isn't enough space for a full sequence of animated character sets. So by default we have to use block 1. To tell the computer which video block we're using, we POKE the block code number into location 56576, as line 450 in both programs does. Notice that the code number is the reverse of the block number. Block 0 has code number 3, and block 3 has code number 0. So in both programs, the variable VM is set to the block number; VB is set to VM*16*1024, which gives us the actual starting address of the video block; and when we POKE a number into 56576, we use 3-VM instead of VM itself.

There are some tricks to this. Because location 56576 is doing several jobs at once, we can't just POKE 56576,3-VM. We have to leave all but two bits of 56576 alone, so we POKE 56576,(PEEK (56576)AND 252)OR (3-VM). And to make sure that data is flowing the right direction through the i/o ports, we first POKE 56578,PEEK(56578)OR 3.

You also have to tell BASIC where to find screen memory, so you POKE the *page number* of screen memory into location 648. The page number is the actual starting address of screen memory divided by 256: POKE 648,SB/256. Notice—it isn't the screen memory location code, but the *actual address* that we divide by 256 to get the number to POKE into 648.

This adds still one more complication. If you press RUN/STOP-RESTORE, the operating system

Continued on page 90

Exposé of the VIC & 64 Operating System

By Donald H. Graham

What kind of magic goes on inside the VIC and 64 that keeps everything in order? Who is the traffic cop that makes sure the computer doesn't crash? The answer is, the operating system. Just what is the operating system (or OS)? We'll give you an introductory overview and along the way show you some tricks to make certain programming chores a little easier to tackle.

POWER UP

When you turn on the CPU, a program in the machine's ROM has the responsibility of getting everything under way. Named the *Kernal* by Commodore, the first thing this program does is check if a ROM cartridge is plugged into the expansion port. Actually, the *Kernal* checks for the presence of the characters "CBM80" near the start of the cartridge area of memory. The cartridge locations start at 32768 on the C-64 and 40960 on the VIC. If a cartridge is present, this "CBM80" code is where it belongs, and the *Kernal* turns over complete control of the CPU to the cartridge. The CPU will then do whatever the cartridge tells it to. In the case of a game cartridge, the BASIC interpreter never gets activated. The CPU just follows the instructions supplied by the cartridge (in machine language). In the case of a language extension like Super Expander or *Simons' BASIC,* the cartridge first lets the OS know it is present, then passes control back to the OS for continuation of the balance of the normal start-up routines. This way the BASIC interpreter *and* the cartridge are both active.

Lots of things happen during the normal BASIC power-up. Some of the tasks that the *Kernal* performs include:

1. Initialize all I/O channels including the serial bus.
2. Start the system clock at 000000.
3. Set up the keyboard scan.
4. Clear the SID chip.
5. Non-destructively test RAM and set the bottom and top of free RAM.
6. Set up the screen in RAM.
7. Initialize zero page pointers, OS working storage, and the cassette buffer.
8. Set default I/O devices, jump vectors, and initialize the screen editor.

This is certainly not a complete list of *Kernal* chores, but it gives you a pretty good idea of how busy the *Kernal* is during that one-to-two-second delay after you hit the power switch or reset button. You can time it if you like. On the VIC it will take a little longer if you have expansion memory in place. Remember, the *Kernal* performs a RAM test which starts at the beginning of memory and continues to the highest available location. Once the system is up, you can simulate the power-up routine with the command SYS 64802 for the VIC, or SYS 64738 for the 64. If you have a program in memory when you execute this command, you don't really lose the program. It's still resident, you just can't get to it easily. There are ways, though.

WHAT NEXT???

Okay, you have the system up, what good is the OS now? Well, you can bet that if the OS is needed to start things up, it's going to be needed to keep things going. Sixty times every second the OS performs these routine tasks:

1. Updates the system clock (found at locations 160, 161, and 162).
2. Scans the keyboard to see if any keys have been pressed (location 198).
3. Updates the screen (locations 1024 through 2023).
4. Keeps track of where the cursor is (locations 201 through 214).
5. Checks for the <RETURN> key (indicating either a command or a program line).

When the OS detects a <RETURN> key, it moves the current screen lines (up to 88 characters on the VIC and 80 characters on the 64) to the BASIC input buffer (locations 512-600). Incidentally, the OS keeps track of how long the current line is (location 213). Once the line is in the BASIC input buffer,

the OS checks the beginning of the line for the presence of a line number. If a line number is there, the typed program line is added to the program in its proper position in RAM. If it doesn't start with a number, the line is executed immediately, one command at a time.

HOW DOES IT KNOW...

How does the CPU know what each command means? Each time the OS encounters a command it takes the current command to the interpreter, which is like a dictionary of the computer's commands. Every time the CPU comes across a BASIC command it must go to the interpreter and look up the command to see what it is. It simply compares the command against a list of legal BASIC keywords, one at a time. It just keeps going down the list till it finds a match or no match. No match results in the well known ?SYNTAX ERROR. When a match is found the OS is directed to the appropriate machine language routine elsewhere in ROM which corresponds to that particular command. The target ML then executes the BASIC command. It doesn't sound too complex, but the number of actual operations being performed by the OS in so short a time is mind-boggling! Figure 1 is a flow chart which, while kind of oversimplifying matters, should give you a clear idea of the process.

As you can see by following the flow chart, the OS analyzes the line you type in and decides what to do with it. If it decides to try to execute the line, it runs to the interpreter to see what the command is. If it doesn't find a match, it sends you an error message. If it decides that it is reading a program line, the OS simply inserts the line into the program in its proper number sequence, *without checking the*

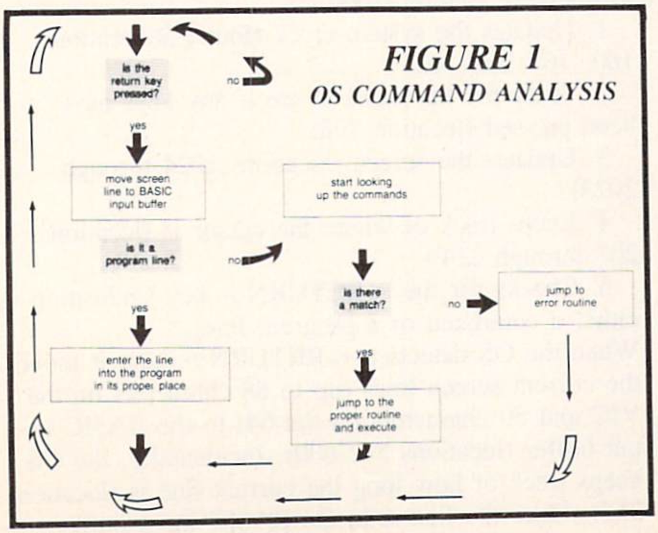


FIGURE 1
OS COMMAND ANALYSIS

legality or syntax of the line! This is an important point. Some systems check command syntax when the program line is entered. Commodore doesn't. You can type any old thing you want in a program line, as long as you don't ask the CPU to try to execute the nonsense. The test for syntax of a program line comes when the CPU tries to execute that command. On the other hand, if the line isn't a program line, the CPU *immediately* tries to execute the line, one command at a time; hence the term "immediate mode" of operation.

WHY BOTHER?...

We've discussed the operating system and some of the chores it performs for us. Now let's take a look at how we can use some of this technique and information. The following is a list of some of the more useful locations that the OS maintains which a programmer can also use to his/her advantage.

Location 19 – Normally contains a zero which sets up a parameter for the INPUT routine. If you POKE anything other than a zero here, you will eliminate the question mark from the INPUT prompt, in addition to the carriage return also associated with this command.

Locations 43 and 44 – Contain a hex representation of the first memory location that BASIC uses (in lo-byte, hi-byte format). If you want BASIC to start somewhere else, you can simply change the values in these locations accordingly (and be sure that the first memory location of BASIC contains a zero).

Locations 45 and 46 – Contain a hex representation of the beginning of the area where your variable values (numbers) will be stored. This is also the end of the current BASIC program. If you want to pass variables from one program to another you must consider the following restrictions:

a) If the first program is larger than the second, you should have no trouble passing variables. Instead of RUNning the second program, use the command SYS 50830. This sets program execution at the beginning of the program *without* clearing out the variables.

b) If the first program is smaller than the second, when you start up the first program the first thing you must do *before* assigning any variable values is to make this vector point *beyond* where the end of the second program will be.

Locations 47 and 48 – Contain a hex representation of the beginning of the area where your variable arrays (numbers) will be stored.

Locations 49 and 50 – Indicate where the end of

your variable array storage is.

Locations 51 and 52—Indicate where the bottom of your string variable storage area is. This area starts at the top of BASIC memory and moves downward.

Locations 55 and 56—Point to where the top of usable BASIC memory is. If you have some machine language that you want to hide somewhere, you can move down the top of memory by adjusting this pointer, then put your ML right above this location, thereby protecting your ML routine from BASIC.

Locations 63 and 64—Indicate the current BASIC DATA line number. If anything goes wrong while READing data from your program and you're having trouble identifying where, you can turn to these locations for help.

Location 144—Contains the value of the reserved variable ST (the STATUS variable). This variable holds a key to the most recent I/O operation. Certain bits set in this location mean certain things. For example, if ST=64, an "end of file" condition has occurred.

Location 152—Contains the number of currently OPEN files.

Location 153—Contains the value of the current INPUT device (normally 0 for the keyboard).

Location 154—Contains the value of the current OUTPUT device (normally 3 for the screen). The I/O device numbers stack up like this:

0: keyboard
1: tape drive
2: RS232 port (through user's port)
3: screen
4: printer
5: printer
6: other serial devices
7: other serial devices
8: disk drive
9: to 31 other serial devices

Locations 160-162—Contain the hi, med, and lo bytes of the internal "jiffy clock" of the machine. The OS uses these values to determine how long it has been since the CPU has been powered up, among other things.

Location 186—Contains the number of the current device number (1 for tape, 8 for disk, etc.)

Location 197 or 203—Contains the value (not an ASCII value) of the key currently being held down. When no key is being pressed, this location contains a 64. If you want to wait for a certain key to be pressed, you can keep testing this location till your condition is satisfied. To wait for the <RETURN> key, just use this line in your program:

```
500 IF PEEK(197) <> 15 THEN 500
```

Location 198—Contains the number of unprocessed characters in the keyboard buffer. If you want to clear the buffer, just POKE 198,0.

Locations 204 through 214—Contain information about the cursor (where it is, if the blink is on or off, etc.).

Location 216—Keeps track of whether the requested "inserts" have been used up. POKE 216,0 and you'll turn off the "insert" mode.

Locations 243 and 244—Point to the current screen color memory location.

Location 631-640—The keyboard buffer. Every time you press a key, it must be processed through this buffer before it's acted on. Proper use of this location along with location 198 can trick the CPU into thinking that you typed something in from the keyboard when you really didn't. The entries are processed on a FIFO (first in, first out) basis. To simulate a cursor to home operation, right before you end your program you must execute this line:

```
60000 POKE 198,1: POKE 631,19
```


Reader Service No. 81

When the program ends, the OS will look at 198 to see if there is a keyboard entry to be processed. It will find one, get that entry from location 631, and process it (the cursor will go to the HOME position).

Locations 641 and 642 – Point to the start of memory.

Locations 643 and 644 – Point to the end of memory.

Location 646 – Contains the value of the color to use when PRINTing to the screen.

Location 648 – Points to the page of RAM where the screen starts.

Location 649 – In this location you will find the size of the keyboard buffer, normally 10. You can turn off the buffer by POKEing a 0 here, and turn it back on by POKEing a 10. If you value your sanity, don't POKE anything larger than 10 into this location.

Location 653 – Keeps track of which shift pattern is currently being used. 1=shift; 2=Commodore key; 3=Shift + Commodore key; 4=Control key; 5=shift + control key; 6=Commodore + control key; 7=shift + Commodore key +control key. This location is a little tricky to use, but can yield some interesting results.

Location 657 – With this location you can lock the ability to switch to upper/lower case. POKE 657,0 will allow case switching, while POKE 657,128 will not allow case switching.

Locations 780-783 – Allow you to pass values directly to the microprocessor when you jump from BASIC to a machine language routine using the SYS command. Just POKE 780 with the value you are passing to the accumulator, POKE 781 with the value you are passing to the X register, POKE 782 with the value you are passing to the Y register, and POKE 783 with the value you are passing to the STATUS register. All that's left is to SYS to the routine you are running and these values will be placed into the appropriate registers and your routine will be executed. This is especially useful when you are calling any of the KERNAL routines from your BASIC program.

Locations 788-819 – Contain a series of vectors (pointers) to various routines in the CPU. You can change these values to accomplish some tricky effects. For example, POKE 808,100 will disable the RUN/STOP key. Or POKE, 792,34 and POKE 793,253 and when you press the RUN/STOP and RESTORE keys, the VIC 20 will perform a RESET (just like SYS 64802). To do the same thing on the C-64, POKE 792,226 and POKE 793,252.

There you have some of the more useful OS memory locations. You can do a lot of other things with these and other locations, like insert your own commands into BASIC, move the screen around, intercept the error message routine, and generate your own error messages. These activities require a little more programming skill, including using machine language, but they can be done! Also, I haven't mentioned quite a few locations that you can tinker with which give some weird results because there are just too many possibilities. Go ahead and tinker, though. I'm sure you'll have fun and you may even be able to expand on this article with your findings. In the meantime you're sure to learn more about your machine. You certainly can't do any harm by just POKEing around in the operating system's memory. □

Tape/Disk Transfer

for the C-64 and VIC 20

By Mark Joerger

Hooray! You finally saved up enough money to buy a disk drive. Now all you have to do is sit down and transfer all those great programs you've saved to tape onto disk. One at a time. Hey, this could take hours! You've got at least a dozen cassettes to do. Load and save...load and save...load and save. This sounds like something the computer should be doing all by itself.

The *Tape/Disk Transfer* program uses a small machine language routine to automatically load a program from tape and save it to disk under the same name. The program will transfer BASIC or machine language files and will reload them into the proper memory location. The program will not transfer data files.

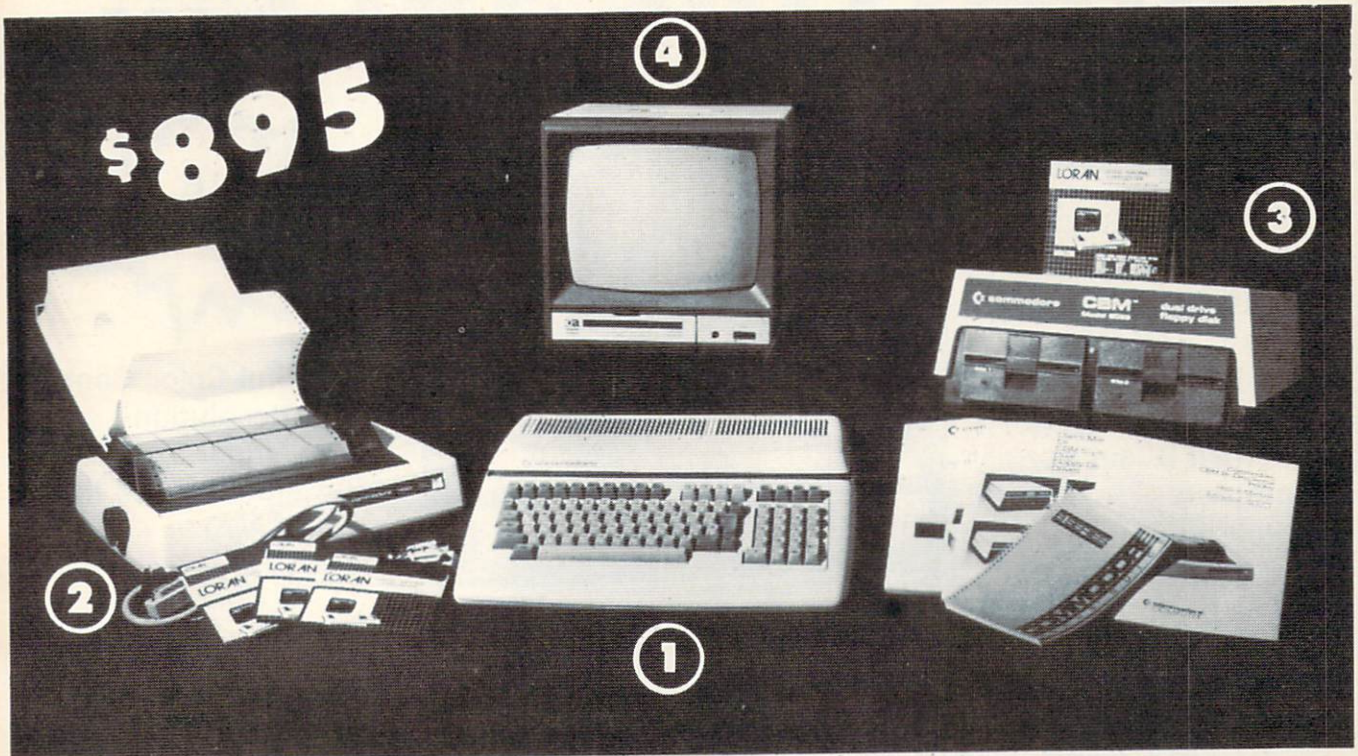
Load and run the program in the usual manner. You will be instructed to place the disk you wish to copy to in the drive and will be given the option of formatting the disk. The program then jumps to the machine language subroutine which instructs you to press play on tape. The first program on the tape will be loaded and saved to disk. The tape drive will then search for the next program from tape and repeat the process. A bad tape load does not crash the program. The search for tape programs will continue to the end of tape. STOP/RESTORE must then be used to regain keyboard control and prepare for the next tape. Disk program names saved with this program will be "padded" out to 14 places with blanks. SYS 53181 (for the 64 version; SYS 7600 for the VIC) will jump back to the machine code routine without reloading the program. ☐

SEE LISTING ON PAGE 85

NEW 128K —MEGA BYTE DUAL DISK DRIVE—80 COLUMN

COMPUTER SYSTEM SALE!

HOME • BUSINESS • WORD PROCESSING



LOOK AT ALL YOU GET FOR ONLY **$895.**

	LIST PRICE
① B128 COMMODORE 128K 80 COLUMN COMPUTER	$ 995.00
② 4023 - 100 CPS - 80 COLUMN BIDIRECTIONAL PRINTER	499.00
③ 8050 DUAL DISK DRIVE (over 1 million bytes)	1795.00
④ 12" HI RESOLUTION 80 COLUMN MONITOR	249.00
• BOX OF 10 LORAN LIFETIME GUARANTEED DISKS	49.95
• 1100 SHEETS FANFOLD PAPER	19.95
• ALL CABLES NEEDED FOR INTERFACING	102.05
TOTAL LIST PRICE	**$3717.95**

$895 SALE PRICE

PLUS YOU CAN ORDER THESE BUSINESS PROGRAMS AT SALE PRICES

	LIST	SALE		LIST	SALE
Professional 80 Column Word Processor	$149.95	**$99.00**	Payroll	$149.95	**$99.00**
Professional Data Base	$149.95	**$99.00**	Inventory	$149.95	**$99.00**
Accounts Receivable	$149.95	**$99.00**	General Ledger	$149.95	**$99.00**
Accounts Payable	$149.95	**$99.00**	Financial Spread Sheet	$149.95	**$99.00**
			Order Entry	$149.95	**$99.00**

PRINTER REPLACEMENT OPTIONS
(replace the 4023 with the following at these sale prices)

	LIST	SALE
☆ Olympia Executive Letter Quality Printer	$699.00	**$379.00**
☆ Comstar Hi-Speed 160 CPS 15½" Business Printer	$779.00	**$469.00**
☆ Telecommunications Deluxe Modem Package	$199.00	**$139.00**
☆ IEEE to Centronics Parallel/Printer Interface	$179.00	**$139.00**

15 DAY FREE TRIAL. We give you 15 days to try out this SUPER SYSTEM PACKAGE!! If it doesn't meet your expectations, just send it back to us prepaid and we will refund your purchase price!!
90 DAY IMMEDIATE REPLACEMENT WARRANTY. If any of the SUPER SYSTEM PACKAGE equipment or programs fail due to faulty workmanship or material we will replace it IMMEDIATELY at no charge!!

Add $50.00 for shipping and handling!!
$100.00 for Alaska and Hawaii orders.
WE DO NOT EXPORT TO OTHER COUNTRIES
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail! We accept Visa and MasterCard. We ship C.O.D. to continental U.S. addresses only.

PROTECTO
ENTERPRIZES (WE LOVE OUR CUSTOMERS)
BOX 550, BARRINGTON, ILLINOIS 60010
Phone 312/382-5244 to order

COMMODORE 64

(with $12.95 Bonus Pack Purchase)

$175.00*

- 170K Disk Drive $239.00 *
- Tractor Friction Printer $169.00*
- 13" Hi-Res Color Monitor $199.00 *

*less coupon discount

COMPUTER AND SOFTWARE SALE

WE HAVE THE BEST SERVICE

WE HAVE THE LOWEST PRICES

SUPER AUTO DIAL MODEM 64

(Best communications package in USA)

$79.00

- Computer Learning Pad $49.00
- New Voice Synthesizer $59.00
- Commodore 64 Power for Vic-20 $69.00

* COMMODORE 64 COMPUTER $175.00

You pay only $175.00 when you order the powerful 84K COMMODORE 64 COMPUTER! LESS the value of the SPECIAL SOFTWARE COUPON we pack with your computer that allows you to SAVE OVER $500 off software sale prices!! With only $100 of savings applied, your net computer cost is $75.00!!

* 170 DISK DRIVE $239.00

You pay only $239.00 when you order the 170K Disk Drive! LESS the value of the SPECIAL SOFTWARE COUPON we pack with your disk drive that allows you to SAVE OVER $100 off software sale prices!! With only $500 of savings applied, your net disk drive cost is $139.00.

* 80 COLUMN 80CPS TRACTION FRICTION PRINTER $169.00

You pay only $169.00 when you order the Comstar T/F deluxe line printer that prints 8½x11 full size, single sheet, roll or fan fold paper, labels etc. Impact dot matrix, bidirectional, LESS the value of the SPECIAL SOFTWARE COUPON we pack with your printer that allows you to SAVE OVER $100 off software sale prices!! With only $500 of saving applied your net printer cost is only $69.00.

* 13" HI-RES COLOR MONITOR $199.00

You pay only $199 when your order this 13" COLOR MONITOR with sharper and clearer resolution than any other color monitors we have tested! LESS value of the SPECIAL DISCOUNT COUPON we pack with your monitor that allows you to save over $500 off software sale prices With only $100 of savings applied, your net color monitor cost is only $99.00. (16 colors).

80 COLUMN BOARD $99.00

Now you program 80 COLUMNS on the screen at one time! Converts your Commodore 64 to 80 COLUMNS when you plug in the 80 COLUMN EXPANSION BOARD!! PLUS 4 slot expander! Can use with most existing software.

80 COLUMNS IN COLOR EXECUTIVE WORD PROCESSOR $49.00

This EXECUTIVE WORD PROCESSOR is the finest available for the COMMODORE 64 computer! The ULTIMATE FOR PROFESSIONAL Word Processing DISPLAYS 40 or 80 COLUMNS IN COLOR or Black and White! Simple to operate, powerful text editing with 250 WORD DICTIONARY, complete cursor and insert/delete key controls line and paragraph insertion, automatic deletion, centering, margin settings and output to all printers! Includes a powerful mail merge.

List $99.00 **SALE $49.00** Coupon $39.00

SPECIAL SOFTWARE COUPON

We pack a SPECIAL SOFTWARE DISCOUNT COUPON with every COMMODORE 64 COMPUTER DISK DRIVE-PRINTER-MONITOR we sell! This coupon allows you to SAVE OVER $500 OFF SALE PRICES!!

(Examples)
PROFESSIONAL SOFTWARE
COMMODORE 64

Name	List	Sale	Coupon
Executive Word Processor	$99.00	$49.00	$39.00
Executive Data Base	$69.00	$35.00	$24.00
20,000 Word Dictionary	$24.95	$14.95	$10.00
Electronic Spread Sheet	$59.95	$49.00	$39.00
Accounting Pack	$49.00	$39.00	$29.00
Practicalc	$59.95	$44.95	$36.95
Programmers Reference Guide	$20.95	$16.95	$12.50
Programmers Helper (Disk)	$59.95	$39.95	$29.95
80 Column Screen (Disk)	$59.95	$39.95	$29.95
Flip & File Disc Filer	$39.95	$16.95	$14.95
Deluxe Tape Cassette	$89.00	$49.00	$39.00
Pro Joy Stick	$24.95	$15.95	$12.00
Light Pen	$39.95	$16.95	$14.95
Dust cover	$8.95	$6.95	$4.60
Pogo Joe	$29.95	$19.95	$16.95
Pitstop II - Epyx	$39.95	$29.95	$26.00*
			*Plus One FREE
Music Calc	$59.95	$39.95	$34.95
Filewriter	$59.95	$39.95	$34.95

(See over 100 coupon items in our catalog)

Write or call for
Sample SPECIAL SOFTWARE COUPON!

EXECUTIVE QUALITY
PROFESSIONAL BUSINESS SOFTWARE

The Cadillac of Business Programs
for Commodore 64 Computers

Item	List	*SALE	Coupon
Inventory Management	$99.00	$49.00	$35.00
Accounts Receivable	$99.00	$49.00	$35.00
Accounts Payable	$99.00	$49.00	$35.00
Payroll	$99.00	$49.00	$35.00
General Ledger	$99.00	$49.00	$35.00

SUPER AUTO DIAL MODEM $79.00

Easy to use. Just plug into your Commodore 64 computer and you're ready to transmit and receive messages. Easier to use than dialing your telephone just push one key on your computer! Includes exclusive easy to use program for up and down loading to printer and disk drives. List $129.00 **SALE $79.00.**

NEW COMPUTER LEARNING PAD $39.95

makes other graphics tablet obsolete. This new TECH SCETCH LEARNING PAD allows you to draw on your T.V. or Monitor and then you can print whatever you draw on the screen on your printers. FANTASTIC!!! List $79.95 **SALE $39.95**

NEW VOICE SYNTHESIZER $59.00

For Com-64 or VIC-20 computers. Just plug it in and your can program words and sentences, adjust volume and pitch, make talking adventure games, sound action games and customized talkies!! FOR ONLY $19.95 you can add TEXT TO SPEECH, just type a word and hear your computer talk—ADD SOUND TO "ZORK", SCOTT ADAMS AND AARDVARK ADVENTURE GAMES!! (Disk or tape).

COM-64 POWER FOR VIC-20 $69.00

Just plug in our 32K RAM MEMORY EXPANDER and you get as much usable programming power as the Commodore-64 computer!! Master control switches on cover. Gold Edge connectors, five year warranty (FREE $29.95 CARTRIDGE GAME).

FLOPPY DISK SALE .98¢

Lowest prices in the U.S.A.!! Single sided, single density, with hub rings, quality guaranteed! (100 bulkpack .98¢ ea.) (Box of 10 $12.00)

COM-64 4 SLOT EXPANSION BOARD $39.95

Easy to use, switch selectable, reset button and LED indicator — saves your computer and cartridges. List $79.00 **Sale $39.95** Coupon $36.95

9" GREEN SCREEN MONITOR $69.00

Excellent quality SANYO, easy to read, 80 columns x 24 lines. Green Phosphorous screen with anti-glare, metal cabinet! Saves your T.V. PLUS $9.95 for connecting cable. Com-64 or VIC-20.

12" GREEN OR AMBER MONITOR $99.00

Your choice of green or amber screen monitor top quality, SANYO 80 columns x 24 lines, easy to read, anti-glare, faster scanning! PLUS $9.95 for connecting cable. Com-64 or VIC-20.

PHONE ORDERS
8AM - 8PM Weekdays
9AM - 12N Saturdays

• LOWEST PRICES • 15 DAY FREE TRIAL • 90 DAY FREE REPLACEMENT WARRANTY
• BEST SERVICE IN U.S.A. • ONE DAY EXPRESS MAIL • OVER 500 PROGRAMS • FREE CATALOGS

Add $10.00 for shipping, handling and insurance. Illinois residents please add 6% tax. Add $20.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. Canadian orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES.

Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail!

VISA — MASTER CARD — C.O.D.

PROTECTO
ENTERPRIZES (WE LOVE OUR CUSTOMERS)

BOX 550, BARRINGTON, ILLINOIS 60010
Phone 312/382-5244 to order

SCUTTLEBUTT

Continued from page 12

VIC AND 64 GAMES

Escape from Infinity Software keeps the fate of the world hanging in the familiar balance while you look for a way off the alien ship onto which you've been shanghaied. On C-64 diskette; $25.00.

Infinity Software, 536 Curie Drive, San Jose, CA 95123 (phone: 408-629-6208).

For what it's worth at the cash register, CBS Software has our respect. Rather than release another spaceship shoot-'em-up, they utilized a previously untapped theme in *Halftime Battling Bands.* Children aged 6 and up must create marching tunes and band formations to stage a halftime show, while trying to trip up the opposing band. C-64 disk or cartridge is $29.95.

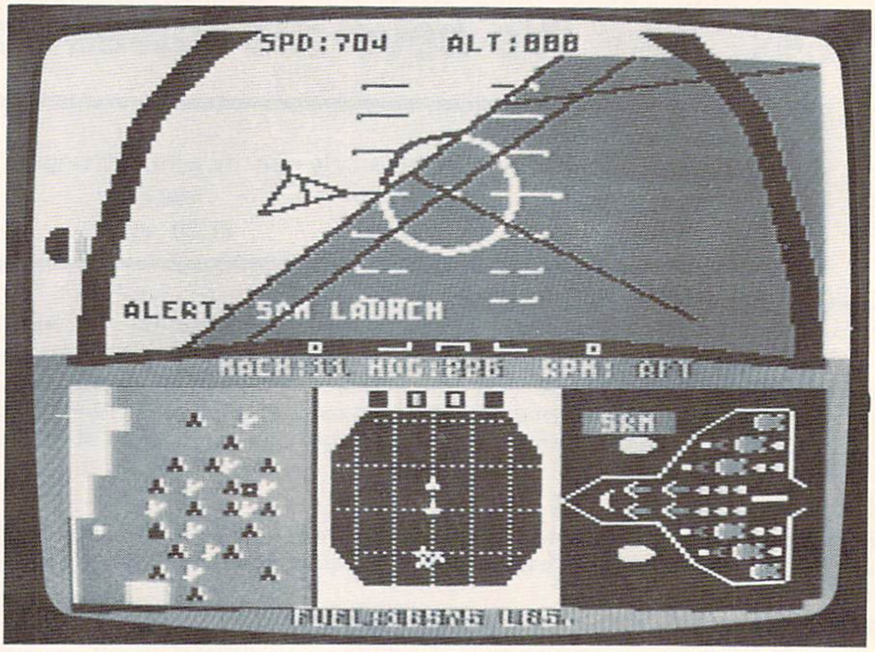
Also from CBS is *Light-Waves,* in which players aged 8 up guide light toward targets by manipulating beam configurations and changing the waves' directions. C-64 disk or cartridge is $39.95.

CBS Software, One Fawcett Place, Greenwich, CT 06836 (phone: 203-622-2500).

Compuzzler is the first release from Uptown Software, a company dedicated to producing "social computer games for intelligent adults." One to four players (or teams) compete to fill in an on-screen crossword, each player advancing from a different corner towards the center and building only upon his own words. As the players meet in the middle, blocking strategies come into play. The C-64 disk contains 70 easy, intermediate, and advanced puzzles, with supplementary disks available. Price is $39.95.

Coming are *Uptown Double Crostics,* a computerization of the famous puzzle requiring the player to use isolated clues to complete a familiar quote, and *Trivia,* with 3600 questions in a multiple choice format. Both for the 64.

Uptown Software, 268 Summer St., Suite #3, Boston, MA 02210 (phone: 617-423-2304).



F-15 Strike Eagle ***includes authentic flight weapons and information systems.***
READER SERVICE NO. 82

If you love combat simulations, you'll want to bayonet yourself a copy of the February *Ahoy!,* with B.W. Behling's *Futurewar!* But if you love them so much that you can't wait another 30 days, several new C-64 releases may quench your thirst for battle.

Microcomputer Games' *Panzers East!* puts you in the enemy's leather boots as you command Axis land and air forces on their invasion of Russia. On disk or cassette for the C-64; $30.00.

Microcomputer Games, Inc., 4517 Harford Road, Baltimore, MD 21214 (phone: 301-254-9200).

Back on the side of the angels, *F-15 Strike Eagle* simulates the experience of flying the modern high-tech marvel. For the 64; $39.95.

MicroProse Software, 10616 Beaver Dam Road, Hunt Valley, MD 21030 (phone: 301-667-1151).

Sonar Search re-creates the struggle for control of the North Atlantic shipping lanes during World War II. With individual screens for each of his three destroyers, the player goes on a hunt for five enemy subs. Keyboard template and 20-page manual are included. On disk or tape; $29.95 plus $2.00 postage.

Signal Computer Consultants, P.O. Box 18222, Pittsburgh, PA 15236 (phone: 412-655-7727).

Four simulations—two military, two civilian—from SSI:

Breakthrough in the Ardennes compresses the Battle of the Bulge into twelve 24-hour turns. Historic strengths of land and air forces are taken into account. Hi-res color map displays the battlefield and its road network. On disk; $59.95.

Broadsides sends you into Napoleonic-style naval conflict

aboard such ships as the Constitution, Guerriere, Bonhomme Richard and the Serapis, each one rigged, manned, and armed just like the original. Price is $39.95.

President Elect lets you recontest any election from 1960 to 1984, with a different outcome possible depending on how successfully the one to three players manage their campaigns during the nine weekly turns from Labor Day to election night. The political climate of the year chosen is a factor, especially with candidates that the user creates by defining their stand on such issues as ERA, arms control, and the PLO. Disk; $39.95.

Computer Quarterback includes pro and semi-pro versions, 36 offensive and 24 defensive options, and computer analysis of playing style. Price is $39.95.

Strategic Simulations Inc., 883 Stierlin Road, Bldg. A-200, Mountain View, CA 94043-1983 (phone: 415-964-1353).

In *Codename: Deadzone*, the mad leader of a hostile nation is about to unleash a deadly virus on the world. You must raid an underground complex to rescue the scientists who have the formula for the antidote. On cartridge for the unexpanded VIC 20; $39.95.

Tymac Controls Corporation, 127 Main Street, Franklin, NJ 07416 (phone: 201-827-4050).

Warlock's Castle challenges you to invade the long-dead magician's fortress, solve clues, avoid dangers, and come out with the hidden treasure. On tape for the VIC 20 with minimum 8K expansion; $8.95 plus $1.00 shipping.

Reilly Associates, P.O. Box 17144, Rochester, NY 14617.

In BitCards' *A Christmas Adventure*, Santa Claus is *not* coming to town—not unless you can unravel the mystery of his disappearance just hours before his annual gift run. You must search the many rooms of Santa's ice castle for clues, find him, and free him. C-64 disk is $14.95; for $2.00 more you get a customization routine that puts your name in several places in the program. For $17.95, BitCards will do the customizing. Add $2.25 for postage.

BitCards Inc., 30 W. Service Road, Champlain, NY 12919 (phone: 514-274-1103).

Mastertronic, Ltd., enjoying a stranglehold (40% and growing) on the computer game market in the United Kingdom, has formed Mastertronic, Inc., in this country with the aim of repeating their UK success. With a price of $8.99 per disk, US game manufacturers had best realize that Mastertronic represents a mortal threat to their interests.

The first ten releases, for both the VIC and 64, will be *Chiller, The Candidate, Space Walk, Duck Shoot, Squirm, BMX Racers, Mind Control, Dark Star, Magic Carpet*, and *Vegas Jackpot.* Three or more new titles are planned per month.

Mastertronic, Inc., 6649 Odessa Avenue, Van Nuys, CA 91406 (phone: 818-780-9230).

The major wrinkle of *Robots of Dawn*, Epyx' text adventure adaptation of the Isaac Asimov novel, is actual conversations between the player (detective Elijah Bailey) and the otherworldly murder suspects he questions. The twists in the plot vary with each game played. On disk for the C-64; $40.00.

Epyx Computer Software, 1043 Kiel Court, Sunnyvale, CA 94089 (phone: 408-745-0700).

GAME CONTESTS

If you're a good enough secret agent to defuse the deadly Buzz-

Bomb, thereby winning *Agent USA*, you're eligible to win a trip via Amtrak for you and two family members to Washington, D.C., where you'll tour intelligence headquarters and the White House. Send Scholastic Software the message that appears on the screen at game end, along with your winning time, well in advance of the April 30, 1985 drawing.

Players who win *Bannercatch*, by piecing together the face of Max the Master Robot, can compete in a drawing for a set of programmable talking robots.

Scholastic Software, 730 Broadway, New York, NY 10003 (phone: 212-505-3000).

All you need do to enter the "Go Mad! For Spy vs. Spy" contest is fill out and mail an entry blank available from participating dealers of First Star Software's *Spy vs. Spy* game before December 31, 1984. Prize is a trip (by plane) to New York for three all- expense-paid days and two all-expense-paid nights, dinner with the staff of Mad, and your caricature in a future issue of the magazine. (Who knows? You could wind up replacing Alfred E. Neuman as the permanent cover model.)

First Star Software, 18 East 41st Street, New York, NY 10017 (phone: 212-532-4666).

BIOCHIP STUDY

Those of you who were fascinated by *The Golden Gateway* (February '84 *Ahoy!*) and *Beyond the Golden Gateway* (March '84 *Ahoy!*) and want to know more about biochip technology are directed to Gorham International's 200-page study of the biochip market that some think will reach the billion-dollar level by 1990. The report, certain to be prohibitively expensive for most, will be published in February 1985.

Gorham International Inc., P.O. Box 8, Gorham, ME 04038 (phone: 207-892-2216).

BABYSITTING SERVICE

Unlike its real-life namesake, it will not raid your refrigerator, run up a bill on your phone, or sneak its boyfriend in your back door. *Babysitter Volume 2* contains four programs that teach young C-64 users number and logic skills.

Clarity Clock uses voice instruction to teach time-telling on traditional and digital clocks; *Krazy Kaleidoscope* allows experimentation with color and pattern designs; *Monkey Maze* teaches hand/eye coordination and logic skills; *Nutty Numbers* pits the child against the computer in a logic guessing game.

K-Tel Software Inc., 11311 K-tel Drive, Minnetonka, MN 55343 (phone: 612-932-4112).

Continued on page 86



Big Value, Small Price!

Why is quality software so expensive?

No wonder people keep asking that question — some of the most popular business packages cost $50, $100, $200, or more!

Well, at DATAMOST we not only asked the same question, we did something about solving the problem of over-priced software. We created KWIK-WARE!™ at only $19.95 per package.

KWIK-WARE! is an exciting new line of Commodore 64 software that includes KWIK-LOAD!™ and KWIK-WRITE!™, as well as other soon-to-be-released software products.

KWIK-LOAD! is a menu-driven, disk utility program for the Commodore 1541 disk drive that allows you to load programs, save files, and copy disks over three times faster than with normal Commodore DOS! KWIK-LOAD! also lets you test the operating speed of your drive, edit information on disk by track or sector, rename files, delete files, validate and format disks, print out sectors, and display and print disk directories, FAST! KWIK-LOAD! helps you save time, which is one of the big reasons you bought your Commodore 64.

And it costs only $19.95!



KWIK-WRITE! is a full-featured word processor for the Commodore 64 with all the text-editing functions of more costly word processors. It includes easy-to-use command menus and help screens, cut and paste, search and replace, print preview, underlining, screen displays up to 132 characters wide, and more.

KWIK-WRITE! supports most popular printers, loads and saves files fast with the help of built-in KWIK-LOAD!, includes a KWIK-REFERENCE! card, and is compatible with other KWIK-WARE! products.

Priced at just $19.95, KWIK-WRITE! delivers the quality and power of the most popular word processors, at a fraction of the price!

At DATAMOST, we didn't just ask why software is so expensive. We did something about it.

DATAMOST™ INC

20660 Nordhoff St., Chatsworth, CA 91311 (818) 709-1202

Commodore 64 and 1541 are registered trademarks of Commodore Business Machines, Inc.
KWIK-LOAD!, KWIK-WRITE!, and KWIK-WARE! are registered trademarks of DATAMOST, Inc.

Reader Service No. 84



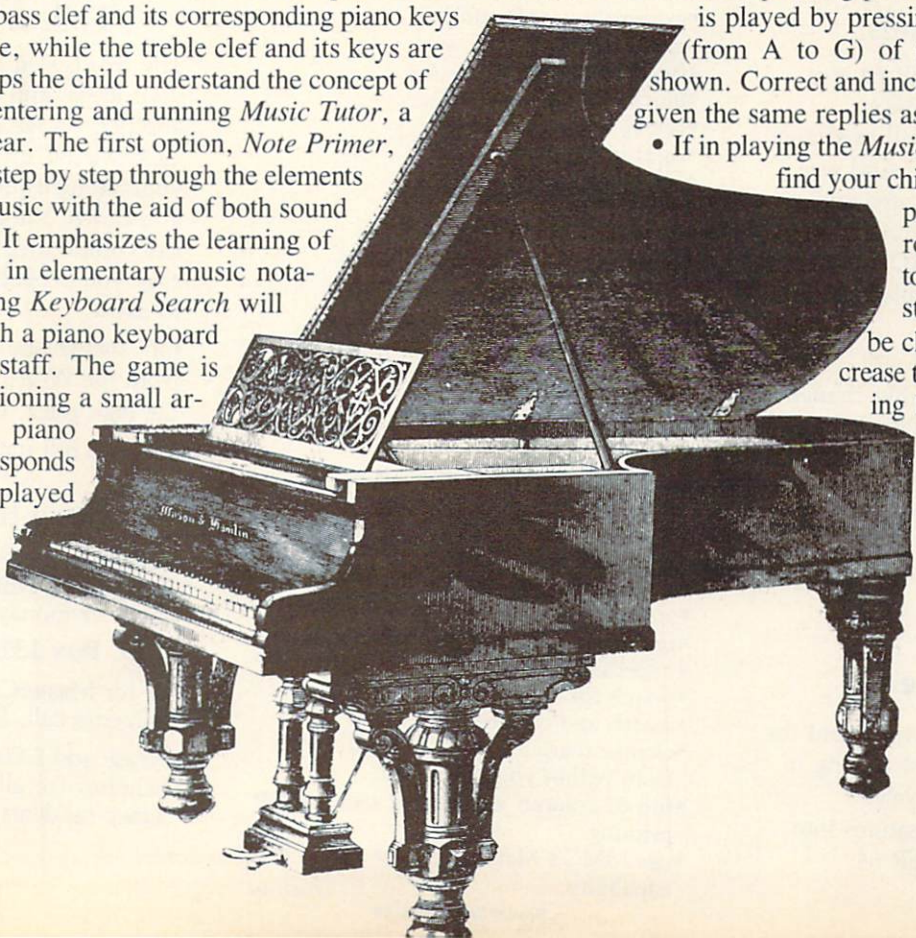
Music Tutor

By Paul Harker

for the C-64

When my daughter began taking piano lessons, she immediately balked at the memorization of music terms and notation. Remembering the names of the keys on the piano didn't come easy either. Thus was born *Music Tutor.* This program turns the usually tedious task of learning music basics into an animated primer and three fun games. Aimed primarily at ages five through ten, it is also excellent for an adult with no music experience. • *Music Tutor* teaches the names of the piano keys and the respective lines and spaces of music notation. An understanding is developed of the relationship between the pitch of a note and its position on the piano keyboard and on sheet music. In both the primer and the games, the bass clef and its corresponding piano keys are colored blue, while the treble clef and its keys are green. This helps the child understand the concept of clefs. • After entering and running *Music Tutor,* a menu will appear. The first option, *Note Primer,* will guide you step by step through the elements of beginning music with the aid of both sound and animation. It emphasizes the learning of the terms used in elementary music notation. • Choosing *Keyboard Search* will present you with a piano keyboard and the grand staff. The game is played by positioning a small arrow under the piano key which corresponds with a note displayed on the staff. A joystick plugged into port 2 is your control. Once the arrow is in position, press the fire button to indicate your answer. An incorrect choice will result in a rude blat sound, while a correct answer will sound the indicated tone and a cheerful clown will appear. The range of notes used in the game can be selected by the function keys. Press f1 for the treble notes, f3 for the bass, and f5 for both. F7 will return you to the menu. • *Pin the Note on the Staff* is the reverse of *Keyboard Search.* Here the object is to position the note on the staff to correspond with the piano key indicated by the arrow. • *Name That Note!* reinforces the learning of the names of the lines and spaces of music notation, and the names of the corresponding piano keys. This game is played by pressing the letter name (from A to G) of the note and key shown. Correct and incorrect answers are given the same replies as the other games. • If in playing the *Music Tutor* games you find your child has difficulty in positioning the arrow or the note due to its speed, the joystick response can be changed. Simply increase the value in the timing loop of line 3010.

SEE PROGRAM LISTING ON PAGE 80



For The C64

"Introducing the Word Processor with Less."



WORD COMMANDER 64: Your Commodore never again at a loss for words.

You have to look very closely to find what we left out of our new word processor. But once you try the WORD COMMANDER 64 you'll discover what's missing. We've taken the *process* out of word processing and put it where it belongs—in our software. With the WORD COMMANDER 64 there's less for you to do and more that it does. And that makes *you* look good.

Only Two Hands.

The WORD COMMANDER 64 is a complete word processor. We've included every function you'll need and some comforts you'll enjoy. And we didn't hide them in complicated codes or commands that require three hands. Everything about the WORD COMMANDER 64 makes plain sense.

Less Hassle.

When you write you probably change your mind a lot. Now changing your text is almost as easy as changing your mind. Like when you want to underline a word or group of words all you do is press "COMMODORE-U" —regardless of the printer control code your printer uses.
Simple.

Less is More.

While simple to use, you won't find the WORD COMMANDER 64 lacking in power or sophistication. We've integrated all of the best features into the WORD COMMANDER 64:

- extremely user friendly
- menu driven
- full word wrap
- block text right
- center text
- right justify text
- delete a block of text
- insert a block of text
- move a block of text
- copy a block of text
- three different fonts supported
- print in bold type
- print in elongated type
- allows sending of any control codes to your printer
- select any background color of your preference
- select any character color as well
- select cursor color
- set top, right, bottom and left margin
- set line spacing
- print headers and footers on every page
- automatically number pages
- print subscripts and superscripts
- underline text
- search for specific text
- search and replace
- change margins, font, or justification from within your document
- full 80 column view of the page before printing
- use MMG's MAIL LIST for mail merge capability

Follow Your Ideas.

The words you need to write represent your ideas; and they flow—one after another. So we designed the WORD COMMANDER 64 to work along with your thought processes. And not slow you down. We did that by making it less complicated and easier to use. So easy you probably already know how to use it.

Oh, there's one more thing missing from the WORD COMMANDER 64, the high price: Only $49.95

Includes full manual and tutorial.

How to Get Yours.

Available at your local dealer or direct from **MMG Micro Software**. Just send check or money order to:

P.O. Box 131 Marlboro, NJ 0774

Or for MasterCard, Visa, and C.O.D. deliveries call: **(201) 431-3472**

Please add $3.00 for postage and handling for all direct orders. New Jersey residents add 6% sales tax.

MMG micro software

Reader Service No. 89

Alice in Adventureland

FOR THE C-64

By Bob Spirko



"I'm late! I'm late!" cried the White Rabbit, glancing at his watch as he ran down the path. Alice stared in wonderment as the well-dressed animal ran close by her. She had never seen a rabbit that carried a watch, let alone one that wore a waistcoat, and so, burning with curiosity, she took off after it.

She followed the rabbit across a field to see it pop down a large rabbit hole. Without thought to the consequences, Alice jumped in after it, only to find herself falling. She fell for what seemed to be a very long time before landing safely (for she fell very slowly). She found herself in a small room with a single opening. Through it, and far ahead, Alice saw the White Rabbit as it disappeared around a corner. But as she started towards the doorway she noticed a small glass table (which certainly was not there before). On it lay a small flat box. "Oh, I can catch up to the rabbit later," she said to herself (although she really was not sure that she could), and went to examine the box. It appeared to be a computer of some sort, as it held a small LCD screen that read "LEVEL 1" and "MAP." Also the outline of a rectangle was displayed, and within it was a tiny white square in the corner. "Oh! That must mean that I am in the corner on the first level," the precocious girl deduced.

Pleased to have such a useful device, Alice set off to find the White Rabbit, unaware of the many dangers and obstacles that lay ahead of her.

As a child I was fascinated by Lewis Carroll's *Alice in Wonderland* and *Through The Looking Glass.* These colorful stories provide an excellent theme for an adventure game. In *Alice,* there is plenty of action, color, and sound as you maneuver through Adventureland, avoiding hazards and searching for gold, keys, and, ultimately, the White Rabbit. Adventureland is actually a complex of six levels and over two hundred rooms, only one of which holds the White Rabbit. RUN the program and you will be asked if this is a new game or a continuation. Press N and a menu appears. Alice is moved by Pressing the Z and X keys to move up and down, and < and > to move left and right. Once a key is pressed, Alice will continue to move in that direction until another key is pressed, an obstacle is encountered, or another room is entered. To end the game press f1. Below the menu, you are asked to select the speed at which you would like Alice to

move. You may wish to go slow until you get used to using the keys in this manner.

Now the screen display is set up. Together, the LEVEL number and the MAP (which is a six-by-six grid) will show you the exact room that you are currently in. Under these are six boxes, empty now, that will hold your keys as you find them. Below this is kept a count of the gold coins that you find. Once the glass table appears you are ready to go. Press X and Alice will enter the north room.

As you explore Adventureland you will encounter a number of rooms bearing various devices:

STAIRS: If Alice enters a room with a staircase going up or down, you will be asked if you wish to use them. If so, press the space bar and Alice will appear in the room above or below. If not, press any other key.

ELEVATOR: You can get to any level you wish by using the stairs but, as the stairs are not aligned, it is a tedious process if you want to ascend or descend more than one level at a time. The elevator is used in the same fashion as the stairs except that after pressing the space bar you will be asked for the level number. Once entered, you will be whisked to the desired level. You need the pink key to operate the elevator.

GREEN DOOR: Most rooms have openings that allow you easy access, but many rooms will have green doors. Normally these impede your progress, but with the green key these will open automatically after you enter the room. (For easy reference, the color of any key matches the color of its device.)

PIT: If you encounter a room with a pit and you are not carrying the blue key, the floor opens up and you will find yourself in the room below. With the blue key, however, you are allowed safe passage.

LOOKING GLASS: The looking glass will transport you to almost any room in Adventureland. Instantly. Unfortunately, you have no choice as to which room. The purple key saves you from this fate.

GRYPHON: So maybe you have collected a few keys and are feeling rather confident. But then there's the Gryphon! If you are in the line of flight of this creature (and there is no escape if you are), you may have a key stolen. The more keys you have, the more likely you are to be relieved of one. The stolen key is then deposited in another room. The Gryphon flies only east to west, so there is no risk if you enter the room from the north or south. You will never lose any key once the yellow key is in your possession.

WHITE DOOR: There are not many of these and only one will lead to the White Rabbit. But you need the white key to get in.

CHESHIRE CAT: This is your ally. Finding a half dozen keys in a couple of hundred rooms rife with dangers and setbacks can be quite a chore. The cat will tell you where to find a particular key, even if it is one that you have lost. You will find the cat on each level, and the key that it mentions corresponds to the level that you are on. On the first level, it will tell you where to find the first key (e.g., YOU WILL FIND THE ELEVATOR KEY IN THE SE SECTION OF LEVEL 5).

GOLD COIN: You do not have to rely on finding a key to possess one. Gold coins appear throughout Adventureland. Whenever you enter a room with a gold coin, it is added to your collection. Once ten gold coins have been collected you will receive a key of the lowest priority not in your possession (see below). If you have all six keys *and* ten coins, the location of the White Rabbit is revealed to you.

RESTAURANTS: When you enter a restaurant you will lose a gold coin. Unlike the keys, lost coins cannot be recovered. As in the real world, there is no safeguard from losing your money once you enter a restaurant.

Keys are arranged by priority:

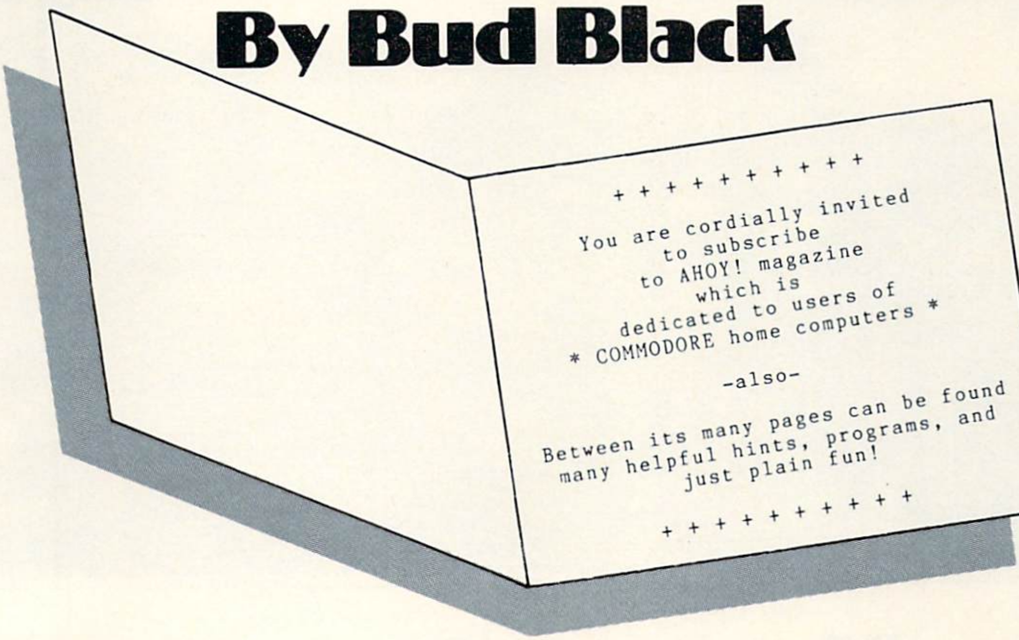
KEY COLOR	DEVICE
1. pink	elevator
2. green	green door
3. blue	pit
4. purple	looking glass
5. yellow	Gryphon
6. white	white door

The first three keys are readily accessible, while obtaining the higher priority keys requires one or two other keys. For example, a high priority key may reside in a room with a green door as its only entrance (although any key that is stolen will be placed in a readily accessible room). If that room currently holds a gold coin, that coin will be displaced.

What makes this game especially interesting is that you will be constantly finding and losing keys, so that you must conduct yourself accordingly. Going from point A to point B with the elevator and looking glass keys will involve an entirely different route than if you had only the keys for the green door and pit, and different again with yet another combination of keys. Having all six keys allows you a great deal of freedom of movement, but you don't

Continued on page 90

MIDPRINT
for the C-64 & VIC 20
By Bud Black



The cost of a party has never stopped me from throwing one. Neither has the knowledge that I'd have to clean up afterward. But the thought of sending out invitations has always sent cold sweat chills down my spine.

Ahoy!'s Ultra Mail program (see page 13) can eliminate some of the drudgery of contacting 10, 20, or 50 or more friends and relatives by putting their addresses on mailing labels. But what about the cards that go in the envelopes? Not only must you buy them at premium prices, but you must then fill each one out individually with the time, date, place, and description of the event. In the computer age? Come on!

The purpose of *Midprint* is twofold. The first option is for printing your own invitations or announcements for special events. The second is for creating general purpose messages such as bulletin board announcements, sales bulletins, and the like. Or you may just want to have fun creating your own epistles.

Each line that is entered (in either mode) will be centered in the middle of the screen. You determine how many words (or special symbols) per line. The result will appear on the screen after entering (ZZZ) and at that point you have the option of printing your result or obliterating it to begin again. This is true in either format.

If you choose just to output to the screen you may wish to experiment by changing print colors. This is easily done since each line entry is no more than an input to a variable string. Of course, if this option is wanted, that particular line should begin with a quote. (This is also true if a comma is desired within the line.)

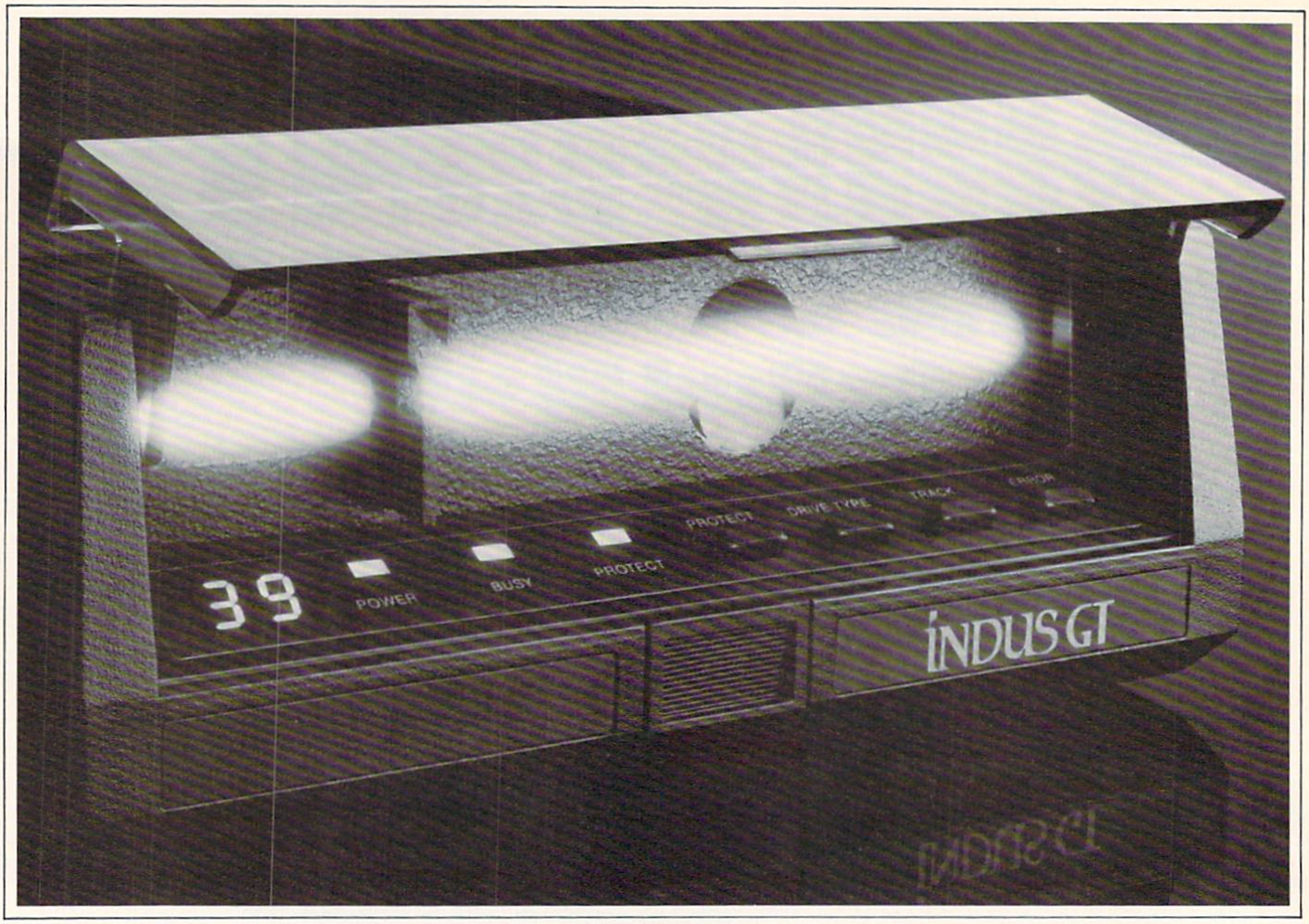
Prior to printing, align the printer paper so that the crease is slightly above the printhead. The program will adjust the paper automatically for the announcement print format.

After your composition is displayed upon the screen, pressing F1 will send it to the printer. F3 will clear the screen and re-initialize the program.

I personally have made good use of this program, as I was recently married and made up my own wedding invitations. I have also used it (with the large print format) to post announcements of coming events on our community bulletin board.

(One modification is needed for VIC use: replace the 20 on line 160 with 11. Note also that the *Bug Repellent* codes provided with the listing are for the 64, not the VIC.) □

SEE PROGRAM LISTING ON PAGE 85



What you get if you cross a Commodore 64 with a Ferrari.

You get the incredible Indus GT™ disk drive.

You get brains. You get beauty.

But, that's not all you get.

You get a disk drive that can handle 100% of Commodore's software. 400% faster.

You get the disk drive with the best service record around. With a one year warranty on parts and labor to prove it.

And, you get the only disk drive that comes with free software. Word processing. Spreadsheet. Database manager. Plus, a carrying case that doubles as an 80 disk storage file.

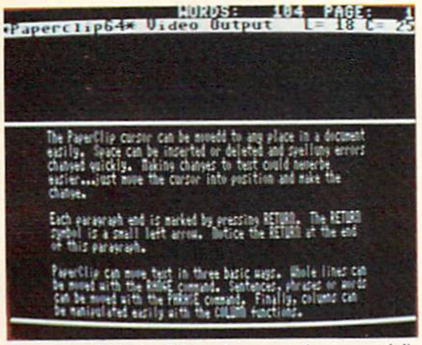
Most of all, you get luxury. From the sleek lines of its soundproofed chassis to the responsive AccuTouch™ controls at the Indus CommandPost.™ From the LED display that keeps you in control of your Commodore to the air-piston operated dust cover that protects your disks and drive.

So, you know what you really get if you cross a Commodore 64 with a Ferrari? You get the best.



INDUS™

REVIEWS



Paper Clip*'s 80 columns (see p. 44).*
READER SERVICE NO. 60

MAGIC VOICE SPEECH MODULE

Commodore Business Machines
Commodore 64; $59.95

The Magic Voice Speech Module provides one of the most realistic speech synthesizers I have ever heard on a micro. This peripheral plugs into the back of your C-64 or SX-64 and is ready for immediate use.

Magic Voice is capable of producing 235 words (or as Commodore likes to call them, utterances). The module also adds four commands to the BASIC interpreter so that you may easily access the speech module:

1) SAY–allows you to tell the module to "say" a word. You may provide the computer with either a string or number. If given a string, the computer will search through its vocabulary until it finds the word, then say it. If given a number, the computer utters the word that corresponds to that number. If given a word or number that does not exist in the machine's memory, an ILLEGAL QUANTITY ERROR? will result.

2) RATE–lets you vary the rate at which the computer says a word. Rate change may vary from .65 slower to 1.4 times faster than normal speech. This is done in order to make speech more natural sounding.

3) VOC–tells the computer when and where an alternate vocabulary exists. Commodore promises that these additional vocabularies will be available soon on disk and will supplement the resident vocabulary. Unfortunately, no mention of how these vocabularies are structured is made in the manual. This information would be helpful to all those hackers who wish to try their luck at forming their own vocabularies.

4) READY–used to read the current status of the speech module. It can tell you three things:

1) The module is not plugged in.

2) The module is busy saying a word.



Put a tongue in your Commodore 64.
READER SERVICE NO. 61

3) The module is dormant awaiting your commands.

Installation of the module is very simple and straightforward. The clear female voice was resounding from my TV within minutes of opening the box.

The manual is clear and easy to understand. It provides numerous examples that are easy to follow, along with full explanations. There's even a section for assembly language programmers, providing all the necessary information for calling the proper routines to drive the speech module from assembly language.

The limited vocabulary has its shortcomings, but is tolerable. With a bit of thought, I was able to piece together practically any phrase with the meaning I sought. Once additional vocabularies become available there should be no problems at all.



Zinderneuf: *death takes a dirigible.*
READER SERVICE NO. 62

Magic Voice should prove a worthwhile investment for those desiring high quality speech from their 64. Commodore states that many of their programs already take advantage of Magic Voice if present, and that many future products will support it as well.

Commodore Business Machines, 1200 Wilson Drive, West Chester, PA 19380 (phone: 215- 431-9100).

–David Barron

MURDER ON THE ZINDERNEUF

Electronic Arts
Commodore 64
Disk; $40.00

Electronic Arts gives you the chance to prove your prowess as a detective in *Murder on the Zinderneuf,* the most intriguing mystery game I have ever played. The screen is a luxury dirigible full with 16 passengers–make that 15 passengers and a dead body. In order to protect your reputation as a super sleuth, you must determine "who done it" before the ship lands in 12 hours (35 minutes in real time). This mystery adventure is cleverly controlled with only a joystick. On the screen, you look down into the passenger compartment of the dirigible as you search the lounge and 16 plush rooms

(almost four horizontally scrolling, high resolution screens).

Murder on the Zinderneuf is a good adventure game for anyone, but for a mystery fan it is a dream come true. The player can choose to be any of eight detectives, each a takeoff on a famous fictional character. You may be anyone from the bumbling Chief Inspector Emile Klutzeau to the charming spinster Miss Agatha Marbles. The real beauty of this variety of choice is that each detective's fictional style is incorporated into the game. As the detective roams the airship, he can question anyone he (literally) bumps into. Using the joystick, you select from a menu of approaches which include confused, seductive, and pushy. The amount and quality of information divulged by each passenger is dependent on your picking the best approach for your detective.

Choose your suspects and informants wisely because you will have time for only a few dozen questions out of the 240 possible combinations. Passenger's rooms may also yield important clues if you are willing to take time out from questioning to search. You may guess who the murderer is when you run out of time. Of course, if you have not gathered enough clues, you will lose even though your suspicions were correct. In any case, you will get to hear the killer's confession and receive a rating from feeble flatfoot to ace detective. If you do solve the mystery, you may still be quite surprised at the killer's statement.

After playing *Murder on the Zinderneuf,* you will undoubtedly become as fascinated with it as I am. Send a self-addressed, stamped envelope to me at 2305 Delgado Drive, P.O. Box 1544, Tallahassee, FL 32303 and you will get back a chart with the key relationships displayed and places to record each clue. Since the game is different each time you play, a second player can provide valuable help with note-taking and deduction.

Electronic Arts, 2755 Campus Drive, San Mateo, CA 94403 (phone: 415-571-7171).

–Richard Herring

PAPER CLIP

Batteries Included
Commodore 64
Disk; $89.95

Belonging to the class of word processor first popularized by *Wordpro* and later enhanced by *Easy Script* (*Ahoy!*, April '84), *Paper Clip* is a logical evolutionary step forward. The similarity of the three WP's is so great that a user familiar with each may occasionally get confused. Unfortunately, for proprietary reasons, the commands for performing identical functions are unique to each program. We suppose that a spirit of cooperation among competing brands is just too much to hope for (if not illegal).

TO WRAP OR NOT TO WRAP

As with many 40-column word processors of this class, text just streams into the buffer. For the most part, the user bangs away without paying attention to the screen layout. The RETURN key is used only to mark the end of a paragraph or to skip a line. All other page formatting is handled by embedded format commands (more on this later). Word wrap is not supported; that is, at the end of a screen line, a word will be split instead of carried to the next line. This feature or lack of it arouses the ire of many reviewers. Word wrap advocates point to the difficulty of proofreading text without it. The opposition points out the confusion generated by the apparent extra spaces placed in the text by word wrap. We have found that, given the inherent restrictions of word processing on a 40-column screen, either approach is satisfactory. What's most important is what the user is accustomed to. If pressed for an opinion, we'd concede that one can do very well without word wrap.

OUTPUT FORMATTING

The appearance of the printed text has only a passing relationship to the entry display. The major part of text output is handled by embedded format commands, or directives. These are quite extensive. Each is designated by the format "checkmark" symbol. No, there is no key on the C-64 which bears this symbol. However, Appendix E of the *Commodore 64 User's Guide* reveals its whereabouts. The shifted "@" key will generate a checkmark when the keyboard is in upper/lower case mode. *Paper Clip* uses the British pound symbol to generate the "checkmark."

This brings us to one of our few minor beefs about the program. The most frequently used functions are the insert mode, the all caps mode, the checkmark, and the command or control key (not necessarily in that order). In *Paper Clip,* these functions are assigned to the Commodore key, the up arrow (not the cursor up), the British pound symbol, and the CTRL key. Batteries Included missed a bet with this assignment. These functions could have been assigned to the function keys on the C-64. This scattering around the keyboard seems to be a carryover from the CBM/PET computers for which *Paper Clip* was originally designed. Instead, the odd

numbered function keys are not used at all, while the even numbered keys handle the screen color assignments.

The wide selection of output format directives may bewilder the beginner. The key is to start off slow, becoming comfortable with the more common commands such as margins, line spacing, page length, and justification. Different commands should then be tried as the need arises. Once the initial concept is mastered, the remainder will follow with little additional effort.

One interesting feature is the ability to create a table of contents file. The "tf" directive sets up the file and the "tb" directive flags each table entry. This feature will be most useful for report writing. Also handy is automatic paragraph indentation.

Single line headers and footers, with automatic page numbering, have their own set of margin and print style controls separate from the body of the text. However, there is no provision for generating on-page footnotes (a common shortcoming).

DISK HANDLING

When it comes to disk operations, flexibility is the word. Although text is normally saved as a program file, the user can select sequential file SAVEs and LOADs. This insures a high degree of compatibility with most other file-generating programs. The disk directory can either be read directly into the text or placed into a separate buffer. A screen read feature allows loading of files from the directory display. A dual disk drive, such as the MSD SD-2 or the CBM 4040, is inherently supported, as the drive number is always requested for disk operations. Single drive users may grow weary of forever entering drive number 0. Since the majority of C-64 users will have one or two single drives, it would have been nice if this feature could be shut off or changed to a second device number. As such, only one single drive is supported.

All standard DOS commands can be issued directly from within *Paper Clip*. The disk error channel can easily be accessed as well.

EDIT MODE

This is the normal operating mode of *Paper Clip* from which all text entry takes place. The top line is reserved for display of control commands, file name, disk operations, and so forth. The second screen line which is highlighted in reverse shows the status of various functions, such as insert mode and caps lock. The right hand end of the line displays the cursor position in terms of line number in the buffer and column number. *Paper Clip* has a capacity of about 511 forty-column screen lines. If this is not sufficient, output files can easily be linked.

Two versions of *Paper Clip* were supplied: the conventional 40-column display and an 80-column display version for use with the Data20 *Video Pak* or the Batteries Included *Super-80 Pak*. Users restricted to 40-column output can set up the display as a horizontally scrolling window on a line up to 250 columns wide. This arrangement is primarily intended to facilitate laying out tables.

Table and chart organization is supported by a set of column-manipulating commands. These allow the setting of text in columns and moving, interchanging, erasing, and duplicating columns. Setting up alphabetized lists is a breeze. Just enter the data in any order and use *Paper Clip*'s sort feature. Numerical data can be sorted as well. Simple column math is provided, the results of which may be placed anywhere in the text.

80-COLUMN VIEW MODE

One of the highlights of this version of *Paper Clip* is an 80-column preview mode. An actual 80-column display (see photo on page 43) is software-generated without need for additional hardware. As a result, each character is only three pixels wide, which limits the readability of the display. The intent, however, is to provide the user with an accurate picture of what the printed text will look like. We found that the Commodore color monitor provided a readable display in this

mode. The display on most color televisions will not be legible.

For users who like to read what is on the screen, *Paper Clip* provides a 40-column preview mode. This sets up the screen as a 40-column window on the output-formatted text. The window can be scrolled horizontally and vertically to view the page format. One nice feature of this mode is the ability to return to edit mode to correct typos or change text. When the changes are completed, output can be resumed at the start of the interrupted page. This feature works for printer output as well as video output.

Both the 80-column view mode and the scrolling window are features recently added to *Paper Clip*. Earlier releases were limited to a 40-column non-scrolling window in view mode.

PRINTER CUSTOMIZING

If you are using a non-Commodore printer, *Paper Clip* is definitely for you. A complete series of printer customizing files are provided for setting up the program to allow maximum use of the available printer features. The user may merge the printer file to create a customized version of *Paper Clip*. Alternatively, the printer file can be called up when you wish to output text. The latter procedure can be particularly useful if you use a high-speed dot matrix for draft copy and a letter quality printer for final form. The distribution disk includes custom files for over a dozen of the most popular printers. These include the Commodore 1525, 1526, 2022, 4022, 6400, 8023, and 8300 printers. Among the other supported brands are NEC, TEC, Epson, Star Micronics, Mannesmann Tally, Okidata, Olivetti, Olympia, and Smith Corona. Complete instructions are provided for creating your own custom printer files.

THE MANUAL

This is one manual where the introductory text, before the table of contents, is of supreme importance. These pages should be carefully read as all the specific details of the Commodore 64 key assignments are included here.

A blessedly brief tutorial chapter gets the user started in short order. The tutorial is complete enough to start up a novice while providing an introduction to all the important features and operations of *Paper Clip*. A very thorough set of reference chapters details all the program operations.

The second half of the manual consists of a set of technical appendices which deal with the operation, customization, and maintenance of *Paper Clip* and its associated disk files. The first appendix is actually an indexed reference chart to all the commands and functions.

THE SECURITY KEY

The distribution diskette can be backed up as the user sees fit. Copyright integrity is maintained by the use of a security key which must be in joystick port 1 when the program is running. Inadvertent loss of text due to removal of the security key is prevented by the program. Simply reinstall the key and pick up where you left off.

The security key gave us some problems. It would not work with the first computer we tried it out with. The key makes special use of the I/O lines at the joystick port. As a result, a computer which functions well with a joystick or paddles may not work with the key. We did get the program running with a second computer. However, after a couple of hours' use, the security key warmed up and became erratic. As a result, the entry of text gradually deteriorated to the point where the program could not be used. A replacement key functioned properly.

The separate *Spellpack* spelling checker ($49.95) and a combined spelling checker version of *Paper Clip* are available. Neither of these were tested for this report.

Batteries Included, 3303 Harbor Blvd., Suite C9, Costa Mesa, CA 92626 (phone: 714-979-0920).

—Morton Kevelson

ADVANCED REPORT GENERATOR

Mirage Concepts, Inc.
Commodore 64
Disk; $49.95

Mirage's *Advanced Report Generator (ARG)* is the companion program to its *Database Manager*

(*DBM*—excuse the acronyms, but they're much shorter), reviewed last month. *ARG* makes up for three critical weaknesses in the report-printing capabilities of *DBM;* it lets you build almost any report format you want, save report formats on disk, and imbed printer commands in your report file. Like *DBM, ARG* is completely menu-driven; the top nine lines always let you know what your options are.

After close inspection, *ARG* has two real shortcomings. First, report formats may not, with limited exceptions, be modified after they are designed. If the format is complex but there is one feature you don't like, you will have to reenter the whole thing. Second, why isn't it included with the *DBM*? Only working together do these two programs provide a full-featured system. (Editor's note: as we went to press, Mirage announced the release of *Database Manager W/Report Generator,* combining *DBM* and *ARG* at a price of $99.95.)

ARG allows you to design three types of report formats—form reports, columnar reports, and mailing labels. At first blush these seem similar to the formats allowed by *DBM.* In fact, each type of report is much more flexible and provides several additional features.

All three report formats allow you to select only those fields you want included on the printout. Each column, in the columnar report, or field, in the form report, can have its own heading; and the entire report can have a heading of up to four printed lines. By comparison, *DBM* form reports force you to use every field, with the label on the original form, in exactly the order of the original form, and with only one report title line. See why you need *ARG*



Vital addition to **Database Manager.**
READER SERVICE NO. 66

as a supplement?

On mailing labels, *ARG* lets you print any fields in any order on as many lines as you want. *DBM* allows only four lines and the first three can contain only one field each. In addition to the calculations included in your original database file, *ARG* lets you add more calculated fields. Up to 50 new text labels and calculated fields can be added to each record from your *DBM* file.

When you design your report format, you can lay it out to fit any size paper your printer will handle. This will allow you to produce professional-looking reports, financial summaries, or invoices. And *ARG* can print as many as 250 copies of one report.

On form reports, *ARG* can automatically total any fields you like. It will just print another form, labeled Totals, with all the other forms' data totaled in the appropriate blanks. When you choose columnar reports, you also have the option of requesting subtotals for any one field.

During the design of your report, you must format each field. This basically means picking one or more options from the onscreen menus. Designate each field as text or numeric, with numeric fields having up to seven decimal places and floating dollar signs if you choose.

Form reports require you to plan the format in detail; actually, they are a pain. Rather than using the Commodore's screen editing capabilities, *ARG* makes you enter row and column numbers for each field and its label. This is supposedly necessary because the C-64 has a 40-column screen display and you're building an 80-column, or wider, printout. I wish Mirage had decided to have the screen scroll horizontally so I could just put stuff where I want it instead of having to chart everything out on graph paper.

And you'd better use graph paper. Designate overlapping fields and *ARG* won't warn you. It will just chop one of them off at print time. Or, design a form with blanks too short for the data and,

once again, any extra characters do a disappearing act when you print.

Your file can be printed in sorted order, if you specify that characteristic in the report and if you created a sort index with *DBM.* Unlike *DBM* reports, *ARG* can print the sorted file in ascending or descending order. If your file is sorted, you also have the option of producing a summary report. It won't give you any data from individual records, but it will print subtotals every time the index (sorted) field changes and totals for the whole file.

My biggest gripe about *ARG* is that once you've built a report format, you cannot modify it. Although they don't provide enough flexibility, there are two exceptions. At print time, you can enter Conditional Statements and Match Field Statements. Both of these let you select only certain records, which meet the criteria you enter, to print. All the BASIC functions, like *less than, greater than,* and *equal to,* are available. Further, you can string several criteria together with *and* or *or* to get a very selective printout of your records.

You'll be able to use most or all of your printer's special features by imbedding print commands in your report. Unfortunately, *ARG* does not give you the required commands, even for the most common printers; you have to dig out your printer manual and find the right appendix.

Even then, *ARG* makes life harder than it needs to be. Since all printer manuals list commands in decimal format, you'd think *ARG* would let you enter your command that way, right? Wrong. You have to compare the command with the ASCII chart in the *ARG* manual and figure out what keystroke that decimal number represents. Decimal 17, for instance, means you have to hit *<control> q.* This is not an insurmountable task, but why can't the program do it?

ARG is a necessary supplement to Mirage's *DBM* if you plan to do any serious work. Given that, I don't know why it's sold as a separate program. Together, the two programs make a useful package. Both have good manuals with outstanding tutorials that introduce you to all their features. But if *ARG* is to stand on its own, it should let you modify existing report formats and design them interactively on the screen. I hate graph paper.

Mirage Concepts, Inc., 2519 West Shaw, Suite 106, Fresno, CA 93711 (phone: 800-641-1441 or 209-227-8369).

—Richard Herring

SC-100 COLOR MONITOR
Sakata U.S.A. Corporation
Commodore 64, VIC 20, Plus/4

Does borrowing the family TV in the middle of *Dallas* cause arguments? Does the poor color rendition or the blur of characters disturb your eyes? Chances are you need a monitor for your computer. The Sakata SC-100 is one of a burgeoning field of Commodore-compatible choices.

Contained in an attractive white case measuring 13.5" high by 14.3" wide by 16.6" deep, the



Audio excellent; video lacks depth.
READER SERVICE NO. 67

unit is a middleweight at 26.2 pounds. Included is a power cord and manual. No assembly is involved.

The 12-page manual is adequate, covering everything from setup to maintenance in a friendly manner by using cartoonlike pictures. There are troubleshooting charts and explanations of all major controls.

The controls are easy to get at, and work exactly as they should. In the front of the unit are the contrast and volume controls, as well as the illuminated power switch and earphone jack. I found this feature useful when working on programs with sound late at night. I could hear what was hap-

pening, but my family was able to enjoy a good night's sleep.

Picture quality is one area in which the SC-100 could stand some improvement. The picture is sharp and the colors are distinct, but they lack the depth I've observed on other units. The audio quality of the SC-100 is excellent, though, rivaling that of any other monitor on the market.

I've tested the SC-100 with the VIC 20, C-64, and Plus/4. It performs best with the Plus/4, providing clean and very sharp colors. With the other machines, a great deal of color bleed occurred.

While offering adequate quality, this general purpose color monitor does not quite match the performance of the 1702 using the chroma-luminescence cable.

Sakata U.S.A. Corporation, 651 Bonnie Lane, Elk Grove Village, IL 60007 (phone: 312-593-3211).

–Shane Duffy

THE MASTER KEY

International Tri Micro
Commodore 64
Disk; $129.00

The Master Key is a software package containing five, count 'em, five programs on one side of a single disk. After loading the main menu, you may select 1) *Cosmic Split*—an action game, 2) *Tri Graph*—a graphics program, 3) *Your Personal Accountant*—a finance package, 4) *Scriptimus*—a word processor, or 5) *Your Filing Cabinet*—a database.

COSMIC SPLIT

In this "terrestrial" game you are required to defend earth from attacking centi—I mean spaceships, as they weave their way down through an ever-increasing field of mushrooms. . .or rather, eggs, while watching out for deadly spiders. Did I say spiders? I'm sorry, I meant to say mutants. Anyway, you get the idea.

To say *Cosmic Split* is a blatant *Centipede* ripoff would be accurate. To say *Cosmic Split* is a very good *Centipede* ripoff, however, is even more precise. The gameplay is virtually identical and the program even features a few new twists.

For example, you may choose to change the color of any or all of the game's elements. Secondly, you may begin on any of ninety-nine difficulty levels and high scores can be saved. If you still enjoy a game of *Centipede* now and then, *Cosmic Split* will come in handy.

TRI GRAPH

As graphics programs go, *Tri Graph* is a real lightweight. Using a joystick-controlled cursor, you can draw in a variety of colors and your choice of nine vertical and horizontal brush sizes. Since a text mode is also available that allows you to label portions of your drawing, *Tri Graph* is suited more to creating graphs and diagrams than artistic masterpieces.

Some useful graphic features that are missing from *Tri Graph* include a magnify mode for detail work, automatic shapes such as circles or squares, and a selection of textured fills such as stripes or checks. The solid fill works well only on the simplest shapes; complex shapes must be filled a section at a time.

If you want a graphics program that will allow you to create simple images and label them, *Tri Graph* may fit the bill. But if you plan on creating state-of-the-art computer graphics, this program won't be of much help.

YOUR PERSONAL ACCOUNTANT

While it's probably true that if you can't balance your checkbook on paper you won't be able to do it on your home computer, a well-written finance program may help you keep track of where your money's going. *Your Personal Accountant* lets you maintain balances on two checking accounts, up to seven charge accounts, and a separate cash account.

You begin by entering your initiation data (account names, most common expenditure categories, beginning balances, etc.). This information is easily corrected or updated and then saved on a separate data disk.

To register a transaction, you must first load the expenditure program and the initiation data. You then enter the month of the

Continued on page 97


James Regan 1985

SPRITE BASICS

A FIRST LOOK AT PROGRAMMING SPRITE GRAPHICS

BY DALE RUPERT

The Commodore 64 contains a powerful chunk of silicon called the VIC-II chip. This integrated circuit provides the majority of the video and graphics capabilities for the computer. This month we'll have a first look at a very useful function provided by the VIC-II chip: sprite graphics.

If you have tried to use sprites in your programs without much success, you are not alone. Defining all the parameters required by the VIC-II can be quite a confusing chore. The software interface provided by standard Commodore BASIC certainly doesn't help. A program full of PEEKs and POKEs is nobody's idea of a friendly programming environment.

We will look at each step of the task to create sprites, to initialize the VIC-II, and to use sprites in our programs. Taken step by step, the job is really not as difficult as it might appear.

First, let's get an overview. A basic sprite is a graphic block which is twenty-four dots or pixels wide by twenty-one pixels tall. Each of these 504 dots is either on or off, that is, either a foreground color or the background color. We must put into memory a group of sixty-three eight-bit words to define which of the dots are on and which are off. Each bit corresponds to a dot within the sprite.

Once the image of the sprite is stored in memory, we must give the starting address of that memory block to the VIC-II chip. We must provide other information to the VIC-II chip such as the color, the horizontal (X) and vertical (Y) positions of the sprite on the screen, and whether the sprite is currently turned on or off. There are some additional concepts which we will discuss next month, such as multicolor and expanded sprites as well as sprite collisions.

After the VIC-II knows where the sprite data is stored in memory and where we want the sprite to appear on the screen, hardware does the rest to put the image on the screen. To cause the sprite to move, we must specify new screen coordinates for it. To change the color of the sprite or to make it vanish, we must also send that information to the VIC-II chip. We send data to the registers within the VIC-II chip the same way that we store information directly into memory, by using POKEs in BASIC.

MEMORY MANAGEMENT

Before we start poking around in memory, we'd better find some safe places to poke. You are no doubt aware of the devastation possible from placing a single byte into RAM at the wrong location. (Actually the worst that can happen is that you might have to turn the computer off and start again. However, two hours' worth of unsaved typing being dumped into the bit bucket is not exactly painless.)

An eight-bit microprocessor with a sixteen-bit address bus can directly access 64K of memory (2 ↑ 16 = 65536 = 64K). The VIC-II can access only one 16K bank of memory at a time. (You can see on the schematic diagram at the back of the *Programmer's Reference Guide* that the VIC-II chip only has fourteen address lines associated with it, A0 through A13, hence a 16K address space.) We will assume that memory bank 0, the default, is selected. All addresses associated with sprites will be contained within

bank 0, ranging from address 0 to 16383.

The data for each sprite image requires a 64-byte *block* of memory (63 bytes for the image and 1 byte as a buffer). Within the selected 16K memory space accessible by the VIC-II are 256 of those 64-byte blocks, numbered 0 to 255. When we tell the VIC-II where the sprite data is stored in memory, we refer to the block number. When we put the data into memory, we must use the actual memory addresses.

The C-64 *Programmer's Reference Guide (PRG)* lists two options for storing sprite data. If you are defining three or fewer sprites, you may store their images in the cassette buffer at addresses 832 to 1022. If you have more than three sprites, you may use addresses from 12288 to 12798. The cassette buffer corresponds to block numbers 13, 14, and 15 (for example, 13*64 equals 832). The other storage area corresponds to block numbers 192 through 199. Just to test your understanding, prove to yourself that the 63 bytes of sprite data stored in block 199 extend from address 12736 to address (199*64 = 12736; note that there are 63 bytes from 12736 to 12798.)

The following programs and discussion assume we are using the storage area with block numbers from 192 to 199. It is possible to use other areas of memory besides these we've discussed, but you must be careful that your BASIC program doesn't interfere with whatever storage areas you choose.

MAKING SPRITES

Now that we have all the difficult stuff out of the way, let's see what it takes to define the shape of a sprite and to store it in memory. The *Sprite Maker* program on page 71 shows one of several ways to create a sprite image.

The twenty-one DATA lines correspond to the twenty-one screen raster lines of a sprite. There are initially twenty-four dots in each line, corresponding to the twenty-four pixel width of the sprite. Once you have typed and saved the program, you may use it repeatedly to create other sprites.

To use the *Sprite Maker* program, list lines 1001 through 1021. Then you move the cursor around and change the dots into any other character, wherever you want the sprite image to be. In the finished sprite, wherever the dots are will be the background color. Wherever you have replaced the dots with any other character, the sprite will display a different color.

One very important rule: after you have created the sprite image within the DATA statements, be sure to move the cursor back up to line 1001 and press the <RETURN> key 21 times! Make your changes in all the lines first, then enter them all into memory as a group. In the program listed above, I have defined a sprite image by replacing the dots with asterisks. Once again, you may create your own images by changing the asterisks back to dots and vice versa.

This program takes the information in the DATA statements one character at a time. The program converts the images into numeric data and stores the information in memory. Each DATA line corresponds to three bytes to be stored in memory. Each character represents one bit.

Each dot corresponds to a bit value of zero, and each "non-dot" has a bit value of one. For example, line 1020 of the program corresponds to the binary number "0001 0000 1110 0000 0000 0000". The leftmost eight bits, when treated as a single binary number, have a value of 16. The middle eight bits have a value of 224 (128 + 64 + 32). The final eight bits have a value of zero.

The three numeric values for each DATA statement are similarly calculated, displayed on the screen, and POKEd into consecutive memory locations. You may specify in which memory block you want the information stored. You may also write down the twenty-four numbers displayed on the screen for future reference. Once you know the numbers, you could write a simpler program to READ and POKE those values into memory. Better yet, let's let the computer do the work for us!

SPRITE SAVER

If you add the lines of the *Sprite Saver* program on page 71 to the previous program, you will automatically save the numeric sprite data values on disk or tape.

These lines create a sequential file consisting of twenty-four sprite data numbers. You may give the file any name up to sixteen characters. Once the information is stored on tape or disk, you may use another program to load it into memory. That way, you can create an external library of sprite images, and you don't have to clutter up your program with sprite data. Each program will simply load whichever sprites it needs as it is executed.

Notice that line 25 will be different for tape storage than for disk storage. The first number in the OPEN statement is the file number, which may be the same for tape and disk. The second number is the device number. Device number 8 refers to the disk, and device number 1 refers to the cassette. The third number is the secondary address. It may be any number from 2 through 14 for disks, so we use 8 for uniformity. For tape, the secondary address tells whether a read or write will be performed. Here, the 1 indicates "write."

The filename that you enter is stored in FL$ and is used in the OPEN statement. The ",S,W" for the disk statement simply indicates that we are creating a (S)equential file and that we are going to (W)rite to it. Notice that all tape files are inherently sequen-

Continued on page 90

COMMODARES

PROGRAMMING CHALLENGES

BY DALE RUPERT

Each month, we'll present several challenges designed to toggle the bits in your cerebral random access memory. We invite you to send your solutions to:

Commodares, c/o *Ahoy!*
P.O. Box 723
Bethel, CT 06801

We will print and discuss the cleverest, simplest, shortest, most interesting and/or most unusual solutions. Be sure to identify the *name* or *number* of the problems you are solving. Also show sample runs if possible, where appropriate.

Your original programming problems would be equally welcome!

Problem #13-1: Starred String

Steven Steckler (Columbia, MD) suggested the following problem. The user enters a string from 1 to 36 characters in length. The computer prints the string with an asterisk border around it. There is one space on either side of the string as the following examples show. If the user types "CAT" or "FLAMINGO," the computer responds with

```
*******          ************
* CAT *    or    * FLAMINGO *
*******          ************
```

Write the shortest program to do this.

Problem #13-2: Auto Run

Mark Ziemba (San Angelo, TX) suggested this challenge. Write a machine language loader which, when run, will load and run another program without the user typing RUN. Any tips, suggestions, and complete programs for this problem are welcome.

Problem #13-3: Micro Calc

Write a program which allows the user to type a sequence of symbols, including any or all of the digits 0 through 9 as well as + and/or −. The computer responds with the statement, "The answer is" and the mathematical result. For example, if the user types "12 + 3 − 2 + 6," the computer responds, "The answer is 19." Is this easier said than done?

Problem #13-4: Data Lister

James C. Dunavant (Gainesville, FL) suggested this *Commodare*. Complete the following program so that the user may select which data items in an array he wishes to view, using only one input statement. For example, if the user types "3-9," data items 3 through 9 are displayed. Just as with the LIST command, entering "3," "-3," and "3-" will display item 3, all items up through 3, and all items from 3 on, respectively.

```
10 REM PROBLEM  #13-4: DATA LISTER
20 REM SUGGESTED BY JAMES DUNAVANT
30 N=20 : DIM D$(N)
40 FOR J=1 TO 20 : D$(J)=CHR$(64+J) :
   NEXT
50 INPUT "VIEW WHICH DATA"; V$
60 ...(YOUR PRGM. STARTS HERE)
```

Several readers sent solutions to Michael Russell's question posed in the September *Commodares* regarding ways to make a program line unlistable. Here's a compendium of the various procedures and suggestions:

• Put a REM <SHIFT-L> in a line ahead of any lines which are to be unlistable.

• Put a REM <CTRL-9> " <SHIFT-M> <SHIFT-S> at the end of any line which is to be unlistable.

• At the end of a line, put a REM <CTRL-9> " and then as many T's as there are characters in the line that you want to be unlistable.

• Ahead of any lines to be unlistable, put REM <CTRL-9> " <SHIFT-M> and press the < back arrow> (above the CTRL key) to change the foreground color to light blue so the following lines are invisible. If your background color is not the default light blue, you must use another key instead of the <back arrow>. For example <SHIFT-P> changes the remaining lines to black.

Thanks to Michael Boivin (Espanola, ONT), Tom Nielsen (Bismarck, MD), Edward Keller (Cincinnati, OH), Eric Spellman (Amarillo, TX), Sean Fletcher (St. Petersburg, FL), Darren Ford (Jacksonville, AL) and Gary Forney (Oelwein, IA) for all of these ideas.

Michael Boivin pointed out that the <RVS-SHIFT-M> causes any control characters following it to perform as if they were in a PRINT statement. This includes RVS ON/OFF, CLR, HOME, cursor movement, and colors, but these happen only during the LISTing of the program.

All of these methods except the first one have one shortcoming. They rely upon erasing or covering up something that is being written to the screen. The problem is that these programs are still listable on a printer.

R.W. Kober (Buffalo, TX) submitted the following ideas which should delight any of you who have worked on this problem:

1. To LIST only line numbers, POKE 774,0. To restore, POKE 774,26.

2. To completely disable the LIST, POKE 775, 168. To restore, POKE 775,167.

3. To produce crazy results, POKE 808,234. To restore, POKE 808,237.

4. To be really mean, cause an unrecoverable system crash with a POKE 775,171. You will have to power-down to recover from this one.

On other topics, Tom Nielsen (Bismarck, ND) submitted the following function in response to *Problem #8-4: Useful Functions.*

```
DEF FNCE(X) = INT(X/2-20)
```

This function is used to center a string of length X on the screen. It is used like this:

```
X = LEN(A$) : PRINT TAB(FNCE(X)); A$
```

Good idea, Tom.

Several readers sent solutions to *Problem #9-1: Passed Words* which indicated that there are more ways than one to break into Marc Spooner's program in the September *Commodares.* John Blair (Huntington Beach, CA) and T. Wayne Biddy (Iroquois Falls, ONT) had identical solutions which were also the simplest way to get into the program. Their solution is:

```
A$ = "XAXX"
B$ = "XXRXXXXX"
C$ = "MXXC"
D$ = "XAXXXXXX"
```

Then press XX and you're in!

Tom Nielsen (Bismarck, ND) and Fred Simon (Alexandria, VA) also sent working solutions. If not the perfect protection system, this problem at least required some serious efforts at "playing computer" to solve it.

There were three distinctly different methods for solving *Problem #9-4: Letter Sorter.* Several readers actually implemented various sorting routines to rearrange the letters of the input sentence. Those programs were longer and more cumbersome than the other two types of solutions.

The following program from Martin Leclerc (St. Hyacinthe, Quebec) goes through the input sentence 26 times, once for each letter:

```
•10 REM PROBLEM #9-4:
•20 REM LETTER SORTER
•30 REM SUBMITTED BY
•40 REM MARTIN LECLERC
•50 INPUT"ENTER A SENTENCE";S$
•60 FOR X=65 TO 90
•70 FOR Y=1 TO LEN(S$)
•80 A$=MID$(S$,Y,1)
•90 IF ASC(A$)=X THEN PRINT A$;
•100 NEXT Y,X
```

The other solution uses an array of 26 elements with each element storing the count of a given letter. Michael Hommer (Waukee, IA) used this method along with the GET statement so that the input sentence could be of any length.

```
•10 REM PROBLEM #9-4:
•20 REM LETTER SORTER
•30 REM SUBMITTED BY
•40 REM MICHAEL HOMMER
•50 DIM. LT(26)
•60 PRINT"ENTER SENTENCE (LETTERS ONLY)"
•70 GET A$:IF A$="" THEN 70
•80 IF A$=CHR$(32)THEN PRINT A$;:GOTO 70
•90 IF A$=CHR$(13) THEN 110
•100 PRINT A$;:A=ASC(A$):LT(A-64)=LT(A-64
 )+1:GOTO 70
•110 PRINT:FOR I=1 TO 26:IF LT(I)=0 THEN
 130
•120 FOR J=1 TO LT(I):PRINT CHR$(I+64);:N
 EXT J
•130 NEXT I
```

These people not mentioned earlier also sent solutions to *Problem #9-3: Common Letters* and/or *Problem #9-4: Letter Sorter:* Barbara Steinman (New York, NY), Long Nguyen (Reno, NV), Dave Antonison (Cambridge, ONT), Tom Dean (Enfield, CT), Kevin Koepsel (Kansas City, MO), Tim Pringle (Flint, MI), Brian Hlubek—age 14 (Clintno, IA), Jim Root (Ann Arbor, MI), Fred Simon (Alexandria,VA), John Blair (Huntington Beach, CA), Michael Boivin (Espanola, ONT), T. Wayne Biddy (Iroquois Falls, ONT), Tom Nielsen (Bismarck, ND), and Vicki Steckler (Columbia, MD). Thanks

Continued on page 90

For The Unexpanded VIC 20

TO THE TOP

As Sparky McSprite, freebooting adventurer who has stolen into the Frizbopian fortress, you have two problems. The first is that you entered on the ground floor and must ascend 32 levels to the top of the fortress and the reward that awaits you there (described below). The second is that Frizbopians don't *like* earthlings in their fortress, and they're going to kill you.

To The Top has four different screens, each with its own assortment of monstrous adversaries, contact with any of whom will have you pushing up Frizbopian daisies. To claim your reward you must pass through all four screens on eight levels of increasing difficulty. I'll briefly explain each screen.

SCREEN 1: BOUNCELDOS' CHALLENGE

Getting to the top in this screen is fairly easy. It's filled with stationary Orkites (boxlike creatures with grim faces) and continuously visited upon by Bounceldos—white LEM-like buggers that descend two at a time from the top of the screen and bounce off the Orkites in their path. The key here is knowing that the Bounceldos always bounce inward: the one on the right bounces left and the one of the left bounces right. This information will come in especially handy when there's a Bounceldo directly above you heading for an Orkite.

SCREEN 2: ORKITES IN ACTION

Remember those boring little Orkites in the first screen that just sat there waiting for you to run into them? Well, they've changed their ways. They now float across the screen in long line formations trying to make you crash into them. And crashing into them is more likely to occur in this screen because it's a Magic Wings level. This means that you automatically thrust upward, with only left-right movement under your control. Also to be avoided in this screen are a few stationary Proppers (red guys with propellers on their heads).

SCREEN 3: BOUNCING GERMIERAGS

Now you are confronted with a screen full of stationary Orkites and three bouncing Germierags that ricochet downward at you every time they hit an Orkite. You're back to normal in this screen—no more defying gravity for now.

SCREEN 4: KINDLY WATCH WHERE YOU'RE FLYING

In this second Magic Wings level you must fly past a bunch of stationary Germierags. You fly at an amazing rate of speed, so I firmly suggest that you be careful. The first few times through this screen will be fairly easy because there are barely any Germierags around. Sometimes you may not even have to move to make it to the top. This is because these screens become progressively harder, and if I hadn't made it as simple as I did the first few times, the highest levels would be nearly impossible.

After you clear the fourth screen the game goes back to the first, which is always slightly harder than the previous time. As for what happens when you reach level 32, you will be awarded (when you finally die) with music and the right to go on playing immortally as long as you wish. This means that whenever you die you will just start over at the same screen—the game will never end. My personal high is level 22, and I programmed this game. I highly respect any of you readers who can get to level 32.

The keys to control Sparky are 'J' to go left, 'K' to go right, and 'F' to go up. You can't go down—it's against your code of honor.

I won't go into any great detail about the program. I will, however, show you its key parts so you can change or modify it at will.

Screen 1: Setup: lines 29-31. Game action: lines 34-44.
Screen 2: Setup: lines 59-61. Game action: lines 63-72.
Screen 3: Setup: lines 9-10. Game action: lines 11-21.
Screen 4: Setup: lines 75-76. Game action: lines 78-83.

If you don't like the characters (although I find them rather cute), you can change the DATA statements at the end of the program. Don't change line 98: that controls the music.

You only have one, man, so play carefully and have fun! □

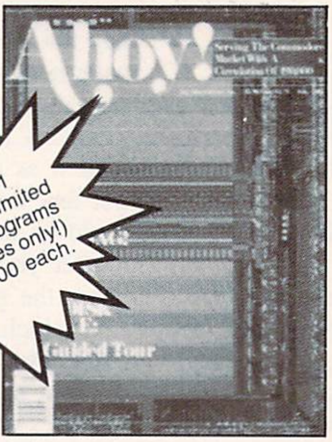
SEE PROGRAM LISTING ON PAGE 83

By KEVIN DEWEY

Back Issues of Ahoy!



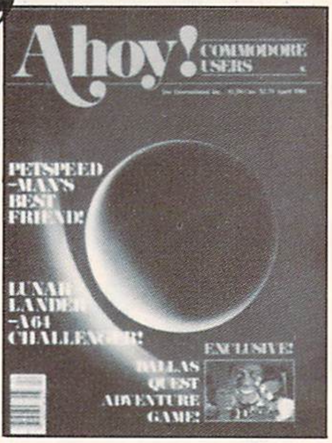
ISSUE #1–JAN. '84 $4.00
64 v. the Peanut! Telecommunications! Tales of the Commodore! And ready to enter: Multi Draw 64 Graphics System! Interrupt Music Maker/Editor! A Peek at Memory! Sequential Files!



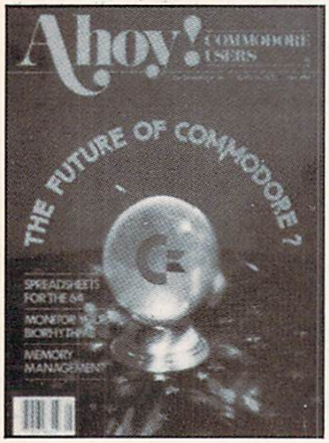
ISSUE #2–FEB. '84 $4.00
Illustrated tour of the 1541! Artificial intelligence! Synapse's Ihor Wolosenko interviewed! And ready to enter: Music Maker Part II! Screen Manipulation! Night Attack! Relative Files!



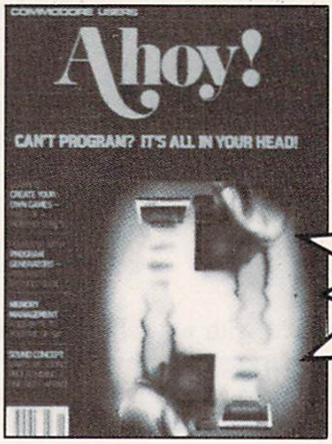
ISSUE #3–MAR. '84 $4.00
Anatomy of the 64! Printer interfacing for VIC & 64! Educational software series begins! And ready to enter: Address Book! Space Lanes! Random Files on the 64! Dynamic Power!



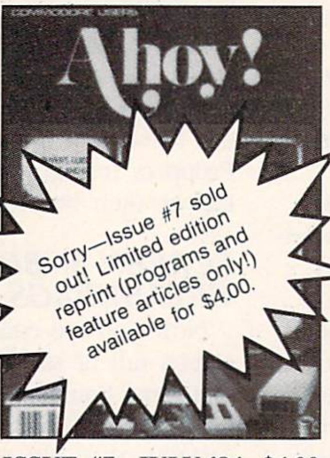
ISSUE #4–APR. '84 $4.00
Petspeed and Easy Script tutorials! Printer interfacing continues! Laserdisc! And ready to enter: Apple Pie! Lunar Lander! Name that Star! Lower case descenders on the 1525!



ISSUE #5–MAY '84 $4.00
Future of Commodore! Inside BASIC storage! Memory management on the VIC & 64! Spreadsheets! And ready to enter: Math Master! Air Assault! Biorhythms! VIC Calculator!



ISSUE #6–JUNE '84 $4.00
Game programming column begins! Program generators! Rupert on inputting! Memory management continues! And ready to enter: Post Time for the 64 & VIC! Alpiner! Sound Concept!



ISSUE #7–JULY '84 $4.00
MSD Dual Disk! Database buyer's guide! Training your cursor! Screen displays! File Sleuth! Users Groups! And ready to enter: Renumbering! Checklist! Math Defender! Brisk!



ISSUE #8–AUG. '84 $4.00
Choosing a word processor! Computational wizardry! Creating your own word games! Sound on the 64! And ready to enter: Micro-Minder! Directory Assistance! Terrible Twins!

ISSUE #9–SEPT. '84 $4.00
Program your own text adventure! Build a C-64 cassette interface! Video RAM! And ready to enter: Salvage Diver! DOS! Sound Explorer! Castle of Darkness! Base Conversions!

ISSUE #10–OCT. '84 $4.00
C-64 Graphics programs! Bit- mapped graphics! Joystick programming! And ready to enter: VIC 40 Operating System! BAM Read & Print! Emerald Elephant of Cipangu! Lawn Job!

ISSUE #11–NOV. '84 $4.00
Music programs & keyboards for the 64! Graphics feature continues! And ready to enter: PTE word processor! Block Editor! Alternate Character Set for the 64! Tunnel of Tomachon!

ISSUE #12–DEC. '84 $4.00
Buyer's guide to printers! 1525 printer tutorial! Custom characters! User Guide to KMMM Pascal! Diving into BASIC! And ready to enter: Construction Co.! Space Patrol! Cross Ref!

Try this experiment! Flip through any issue of *Ahoy!* that you've finished reading. Add up the hours of programming toil our feature articles and tutorials saved you—the hours of shopping our expansive news section saved you—and the money our unbiased reviews saved you—and see if the magazine didn't pay for itself! Then ask yourself: can you afford to be without a single time- and money-saving back issue?

If you're serious about programming, you can't be serious about continuing without a complete collection of *Ahoy!*

Send coupon or facsimile to:
Ahoy! Back Issues
Ion International Inc.
45 West 34th Street—Suite 407
New York, NY 10001

If you have a modem and want more complete information on any of the issues listed, call the *Ahoy!* Bulletin Board. (See page 24.)

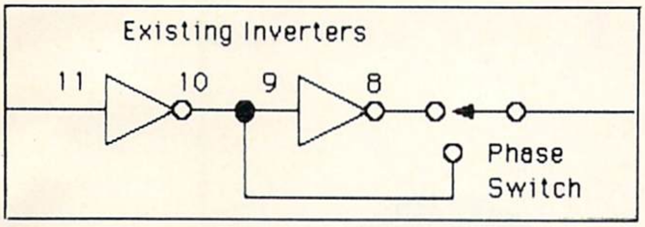
Ahoy!

Please Send Me The Following:
_____ Copies of issue number _____
_____ Copies of issue number _____
_____ Copies of issue number _____
Enclosed Please Find My Check or Money Order for $__________
(Outside the USA please add $1.00 for every copy)

NAME__________________
ADDRESS__________________
____
CITY__________________
STATE__________________
ZIP CODE__________________

FLOTSAM

This month we turn Flotsam *over in its entirety to Bob Kovacs, who sent us an enhancement to Anthony Scarpelli's* Building a Cassette Interface for your Commodore 64 *(Sept. '84* Ahoy!*) that was too good to condense.*



A recommended modification of the original design.

Mr. Scarpelli's article in the September issue of *Ahoy!* reminded me of my own efforts about two years ago. I designed the Universal Tape Interface and Duplicator (UTI&D) which, like Mr. Scarpelli's circuit, connects a conventional tape recorder to Commodore's computers like the VIC 20 and C-64 which have a cassette port using a 6-pin edge connector. My product is currently manufactured, under license, by Tymac (who also manufactures another design of mine: The Connection printer interface).

Using a conventional cassette tape recorder instead of Commodore's Datasette has several advantages. First of all, most folks have a cassette recorder lying around. Second, a conventional recorder allows the use of voice recording (which the Datasette does not). I use the built-in microphone to record a brief statement about the program (or data file) which follows it. For instance, you could indicate the program name, the date, version number, and something about its function or use. With the high speed playback functions found in many recorders (commonly called cue and review) it's possible to quickly scan through a 30 minute tape. Third, since the conventional tape recorder has a speaker, it's possible to actually listen to the program or data. This can be very helpful as a diagnostic in situations where there is a loading problem.

Although Mr. Scarpelli indicated that he was satisfied with the performance of his design, he admits that he hasn't tried it with any other recorders. As a result, I suspect that those who attempt to build one for themselves may experience some problems. It's certainly not easy to design a device like a tape interface that works reliably with any recorder. There are hundreds and hundreds of different recorders, and each one has its own unique characteristics.

The main reason for writing this letter is to suggest two changes in Mr. Scarpelli's design which would make it more universally acceptable.

The relay used in Mr. Scarpelli's interface is of the reed relay variety. This relay is attractive since it is available through many parts suppliers and is also quite compact. The potential problem here is that although this type of relay probably has an adequate sustained current rating, its switching capacity is quite limited. The sustained current for a typical portable tape recorder is in the 100 to 200 milliamp range. The switched current, that is, the transient current pulse that occurs at the instant the switch closes, is substantially greater. Switching currents have been measured to be on the order of 1 amp and as much as 2 amps.

Since the common variety of reed relays have a limited current switching capability (usually ¼ to ½ amp), it's only a matter of time before the contacts become pitted and seize. I chose a heavy duty 5 volt relay in my design (2 amp switching current) to avoid this problem.

The second area of concern has to do with the number of inverters used in the record and playback sections of Mr. Scarpelli's design.

Each inverter inverts, or flips, the signal by 180°. Since the Commodore tape I/O routines are phase sensitive, it is critical to have the right number of inverters for reliable operation. But what is the right number? That depends on the characteristics of the particular tape recorder. It's possible that there are one or more inverters in either the record or playback (or both) circuitry internal to the tape recorder. If that is so, it is necessary to compensate for it in the interface in order to maintain compatibility. I recommend that Mr. Scarpelli's design be modified as shown in the figure. The extra inverter and switch in the playback section can be set as required for the most reliable operation.

Once the phase switch is set it is not necessary to change it, so long as you are saving and loading programs on the same recorder. Loading a program that was saved on a different recorder, however, may require that you flip the switch, as that other recorder may have had an additional signal inversion.

My UTI&D takes a different approach in the processing of audio signals and also has a few extra "bells and whistles" (one reviewer referred to it as the "Cadillac of tape interfaces"). Nevertheless, I believe that Mr. Scarpelli's design is adequate and can be made more reliable with the use of a heavier-duty relay and the phase compensation switch.

—Bob Kovacs
West Orange, NJ

YOUR COMMODORE 64™ CAN NOW USE STANDARD APPLE™ II+HARDWARE AND SOFTWARE



WITH THIS

At Mimic we believe that you and your computer should dictate the choices of hardware and software you can use.

The Spartan™ was developed to allow you to choose the hardware and software that best suits your needs.

Our goal in designing the Spartan™ was simple. To take what you already have and give you more.

Mimic Systems is proud to give you the Spartan™ The Apple™ II+ emulator for the Commodore 64™

Spartan™ Suggested Retail Prices:

The Spartan™ (includes BUSS, CPU, and DOS cards) $599.00
BUSS card $299.00
CPU card (requires BUSS card) $199.00
DOS card (requires BUSS and CPU card) $199.00

(All prices in U.S. Funds. Freight not included.)
American Express, Visa and MasterCard accepted.

Commodore 64 and Commodore logo are trademarks of Commodore Electronics Ltd. and/or Commodore Business Machines, Inc. Apple® II+ is a trademark of Apple Computer, Inc. Spartan™ is a trademark of Mimic Systems Inc., and has no association with Commodore Electronics or Apple Computer, Inc. The Spartan is manufactured by Mimic Systems Inc. under license granted by ATG Electronics Inc. of Victoria, B.C., Canada.

MIMIC

FOR INFORMATION WRITE:

MIMIC SYSTEMS INC.
1112 FORT ST., FL. 6S
VICTORIA, B.C.
CANADA V8V 4V2

To Order Call:
1-800-MODULAR
(663-8527)

Reader Service No. 101

GHOST BUSTERS

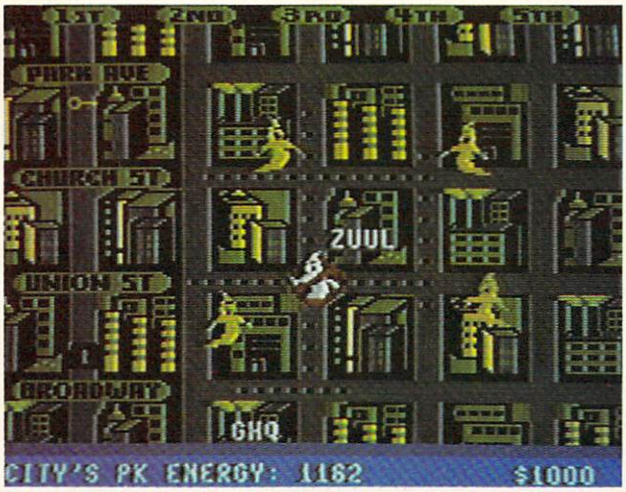
An *Ahoy!* Exclusive

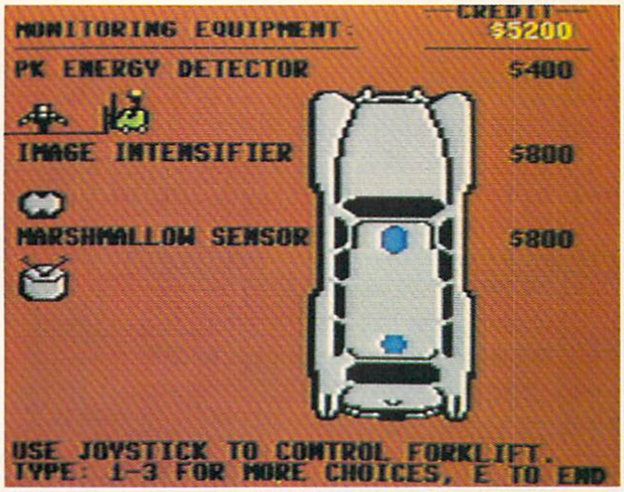
By Robert J. Sodaro

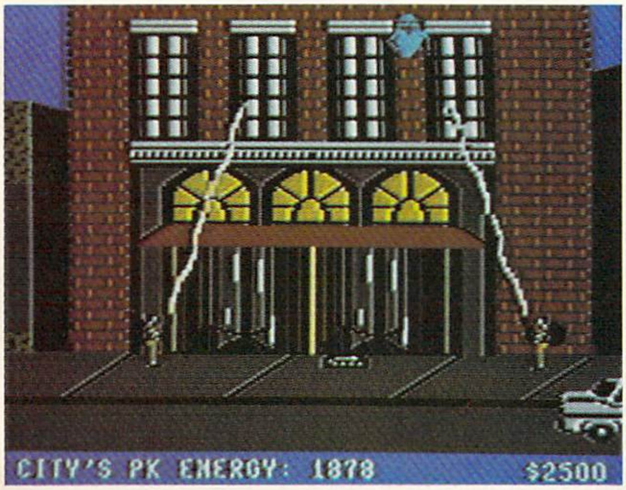
"Who you gonna call?" went the most popular musical question of 1984. Tens of millions of moviegoers answered *"Ghostbusters."* Columbia Pictures stockholders answered "My accountant – my banker – my broker." Columbia's merchandising executives answered "The T-shirt makers – the lunchbox manufacturers – the Saturday morning cartoon producers" as they launched the largest licensing spree for any movie since *Star Wars.*

One of those calls, of course, was to a computer game company. If you want reputation, a hit-laden track record, and some of the nation's top designers, who you gonna call? Activision, of course.

Ghostbusters represents the gaming giant's first experience with licensing. They'd always shunned that path, preferring to lead the market with landmark games like *River Raid* and *Pitfall.* But *Ghost-*









Top left: *spook's-eye view of the city, from which players spot haunted buildings and stray Roamers.* **Top right:** *with a make on a specter, the Busters head toward the scene.* **Bottom left:** *breaking out their portable containment devices to trap a slimer.* **Bottom right:** *can your men toast the dreaded Stay-Puft Marshmallow Man?*
READER SERVICE NO. 105

busters was "the right product at the right time," according to a company spokesperson. In other words —would *you* say no if a fortune walked in *your* door?

The agreement signed, it was Activision's turn to ask themselves—who you gonna call? They decided on David Crane, designer of *Pitfall I* and *II, Decathlon,* and other hits. "I'd just seen the movie, and loved it," David told us. "I was working at the time on a game that had you driving around the city doing something; I hadn't quite figured out what. When I got the *Ghostbusters* assignment I simply adapted my idea to the storyline."

The Commodore 64 was chosen because, said David, "its hi-res graphic capabilities make it the best micro for designing action-oriented games." He went on to praise the superior sound made possible by the SID chip. Blasphemy for one of the most prominent proponents of the Atari 2600, who still designs most of his games on the machine? "I've been quoted as saying that nothing is impossible on the 2600, but this game probably would have been," said David. The *Ghostbusters* program is 19 times larger than *Fishing Derby,* David's first for Activision.

Foremost in David's mind was capturing the essence of the film in the gameplay. "I was especially concerned about building the movie's humor into the game, to make the game as much fun as the movie." In this goal, we think David succeeded. For instance, your men will shout "Ghostbusters!" when they catch a slimer (one of the mischievous sprites of the movie); when downed, they will cry, "I've been slimed!"

Once the game has been booted, it displays the no-ghost logo and (accompanied by onscreen lyrics and a bouncing ball) plays the entire movie theme. From here the player goes into the setup mode, where he acquires a loan to start a Ghostbuster franchise and buys his equipment. This done, the gamer hits the streets. (The game description that follows will, unfortunately, be a bit confusing for those readers who haven't seen the movie. Our apologies to both of you.)

Five main screens are utilized: an overview of the city, displaying streets, buildings, the temple of Zuul, and Ghostbusters' HQ; a closeup of the city streets as you drive around; closeups of various buildings which may or may not be haunted; a closeup of the Marshmallow Man as you attack him; and the entrance to Zuul's kingdom.

You cruise in your Ghostmobile, hunting for roamers and slimers, trapping them and transporting them back to headquarters. As the city's PK energy level rises, more ghosts will appear. When four roamers meet, they join to form the Marshmallow Man. Once formed, he will tromp a building into rubble, again raising the PK level. The higher the PK level, the more powerful Zuul becomes.

There are three possible endings: 1) The Gatekeeper and the Keymaster connect, and you haven't made a profit since the start of the game, causing your world to be absorbed into Zuul's fiery dimension. 2) You're rolling in green when the two demons join, but are unable to sneak two of your Busters past Zuul into his temple, again causing doom and destruction. 3) You successfully sneak past Zuul to win the game and postpone Armageddon. Upon winning, you can sock your extra cash away in an account for use in future games.

A game this complex would normally have taken David over a year to develop, but he brought in the finished product in about six weeks. "I think I had the world's shortest deadline for a game of this type," said David, who had to triple up on his regular three to four hour a day work schedule. He also received "a lot of help" with the designing. "Someone else wrote the music which plays throughout much of the game, and others helped with various aspects. It was a real team effort."

David Crane and Activision deserve special recognition for this one. Not for producing a game that excels in terms of both graphics and gameplay, but for doing it when they didn't *have* to. The power of a name being what it is, any slimy piece of software that bore the name *Ghostbusters* would have been a big stocking-stuffer this Christmas. And that philosophy appears to inform product-licensing in all genres —to produce the worst notebook cover, or Halloween costume, or novelization that the traffic will bear.

We're glad that companies like Activision never heard of that philosophy. If the Motion Picture Academy adds an Oscar this year for Best Computer Game Based on a Feature Film, we know just who the person that opens the envelope is gonna call. ☐

ERRATA

The final line of the main program of the *VIC 40-Column Operating System* (October *Ahoy!*) was omitted during printing. It reads:

:2E68 2A 00 00 00 00 00 00 00

To enter, LOAD the main program, LOAD and RUN the entry program, type in the above line, SAVE the new version, then LOAD and RUN the boot program. (Persons with ML monitors can enter this information directly into the disassembled program.)

Several October program disks shipped out with bogus versions of the program as well. If your disk has an unoperating *VIC 40-Column Operating System,* return it to Double L Software, P.O. Box 1758, Staten Island, NY 10314 for a prompt replacement.

Also in the October issue, the first two lines of the chart at the top of page 16 in *An Overview of Bit Mapped Graphics* should read as follows:

0	3	0	\$0000-\$3FFF
1	2	16384	\$4000-\$7FFF

Finally, an addition to *PTE* in the November *Ahoy!*: to return to the main menu from text, hit the RUN/STOP and Commodore keys simultaneously.

Our apologies for the above errors/omissions.

PROGRAM LISTINGS

As of our November issue we began utilizing an improved method of printing our program listings. This new method will make entering programs easier for you, but only if you read this entire page carefully before proceeding!

Additionally, as of this issue, our *Bug Repellent* line codes will be found in the whited-out area to the right of their respective program lines.

On the following pages you'll find several programs that you can enter on your Commodore computer. But before doing so, read this entire page carefully.

To insure clear reproductions, *Ahoy!*'s program listings are generated on a daisy wheel printer, incapable of printing the commands and graphic characters used in Commodore programs. These are therefore represented by various codes enclosed in brackets [ ]. For example: the SHIFT CLR/HOME command is represented onscreen by a heart. The code we use in our listings is [CLEAR]. The chart below lists all such codes which you'll encounter in our listings, except for one other special case.

The other special case is the COMMODORE and SHIFT characters. On the front of most keys are two symbols. The symbol on the left is obtained by pressing that key while holding down the COMMODORE key; the symbol on the right, by pressing that key while holding down the SHIFT key. COMMODORE and SHIFT characters are represented in our listings by a lower-case "s" or "c" followed by the symbol of the key you must hit. COMMODORE J, for example, is represented by [c J], and SHIFT J by [s J].

Additionally, any character that occurs more than two times in a row will be displayed by a coded listing. For example, [3 "[LEFT]"] would be 3 CuRSoR left commands in a row, [5 "[s EP]"] would be 5 SHIFTed English Pounds, and so on. Multiple blank spaces will be noted in similar fashion: 22 spaces, for example, as [22 " "].

Sometimes you'll find a program line that's too long for the computer to accept (C-64 lines are a maximum of 80 characters, or 2 screen lines, long; VIC 20 lines, a maximum of 88 characters, or 4 screen lines). To enter these lines, refer to the *BASIC Command Abbreviations Appendix* in your User Manual.

On the next page you'll find our *Bug Repellent* programs for the VIC 20 and C-64. The version appropriate for your machine will help you proofread our programs after you type them. (Please note: the *Bug Repellent* line codes that follow each program line, in the whited-out area, should *not* be typed in. See the instructions preceding each program.) □

When You See	It Means	You Type		You Will See
[CLEAR]	Screen Clear	SHIFT	CLR/HOME	
[HOME]	Home		CLR/HOME	
[UP]	Cursor Up	SHIFT	↑CRSR↓	
[DOWN]	Cursor Down		↑CRSR↓	
[LEFT]	Cursor Left	SHIFT	←CRSR→	
[RIGHT]	Cursor Right		←CRSR→	
[SS]	Shifted Space	SHIFT	Space	
[INSERT]	Insert	SHIFT	INST/DEL	
[DEL]	Delete		INST/DEL	
[RVSON]	Reverse On	CNTRL	9	
[RVSOFF]	Reverse Off	CNTRL	0	
[UPARROW]	Up Arrow		↑	
[BACKARROW]	Back Arrow		←	
[PI]	PI		π	
[EP]	English Pound		£	

When You See	It Means	You Type		You Will See
[BLACK]	Black	CNTRL	1	
[WHITE]	White	CNTRL	2	
[RED]	Red	CNTRL	3	
[CYAN]	Cyan	CNTRL	4	
[PURPLE]	Purple	CNTRL	5	
[GREEN]	Green	CNTRL	6	
[BLUE]	Blue	CNTRL	7	
[YELLOW]	Yellow	CNTRL	8	
[F1]	Function 1		F1	
[F2]	Function 2	SHIFT	F1	
[F3]	Function 3		F3	
[F4]	Function 4	SHIFT	F3	
[F5]	Function 5		F5	
[F6]	Function 6	SHIFT	F5	
[F7]	Function 7		F7	
[F8]	Function 8	SHIFT	F7	

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

VIC 20 BUG REPELLENT

By Michael Kleinert and David Barron

The program listed below will allow you to quickly debug any *Ahoy!* program you type in on your VIC 20. Follow directions for cassette or disk.

For cassette: type in and save the *Bug Repellent* program, then type RUN 63000[RETURN]SYS 828[RETURN]. If you typed the program properly, it will generate a set of two-letter line codes that will match those listed to the right of the respective program lines (within the white border). (If you didn't type the program properly, of course, no line codes will be generated. You'll have to debug the *Bug Repellent* itself the hard way.)

Once you've got a working *Bug Repellent*, type in the program you wish to check. Save it and type the RUN and SYS commands listed above once again, then compare the line codes generated to those listed in the magazine. If you spot a discrepancy, a typing error exists in that line. Important: you must use exactly the same spacing as the program in the magazine. Due to memory limitations on the VIC, the VIC *Bug Repellent* will register an error if your spacing varies from what's printed.

You may type SYS 828 as many times as you wish, but if you use the cassette for anything, type RUN 63000 to restore the *Repellent*.

When your program has been disinfected you may delete all lines from 63000 on. (Be sure the program you type doesn't include lines above 63000!)

For disk: type in the *Bug Repellent*, save it, and type RUN:NEW[RETURN]. (See above regarding testing the *Bug Repellent* on itself.) Type in the program you wish to check, then SYS 828. This will generate a set of two-letter line codes that you should compare to those listed in the magazine to the right of the respective program lines (within the white border).

To pause the line codes listing, press SHIFT. To permanently pause it, press SHIFT LOCK. To continue, release SHIFT LOCK.

To send the list to the printer type OPEN 4,4:CMD 4:SYS 828[RETURN]. When the cursor comes back, type PRINT#4:CLOSE 4[RETURN].

```
·63000 FOR X = 828 TO 1023 :READ Y:POKE X
 ,Y:NEXT:END                                MH
·63001 DATA 169, 0, 133, 63, 133, 64, 165
 , 43, 133, 251                             BD
·63002 DATA 165, 44, 133, 252, 160, 0, 13
 2, 254, 32, 228                            FO
·63003 DATA 3, 234, 177, 251, 208, 3, 76,
  208, 3, 230                               ND
·63004 DATA 251, 208, 2, 230, 252, 169, 2
 44, 160, 3, 32                             DJ
·63005 DATA 30, 203, 160, 0, 177, 251, 17
 0, 230, 251, 208                           LP
·63006 DATA 2, 230, 252, 177, 251, 32, 20
 5, 221, 169, 58                            JB
·63007 DATA 32, 210, 255, 169, 0, 133, 25
 3, 230, 254, 32                            JF
·63008 DATA 228, 3, 234, 165, 253, 160, 0
 , 170, 177, 251                            KA
·63009 DATA 201, 32, 240, 6, 138, 113, 25
 1, 69, 254, 170                            HP
·63010 DATA 138, 133, 253, 177, 251, 208,
  226, 165, 253, 41                         KJ
·63011 DATA 240, 74, 74, 74, 74, 24, 105,
  65, 32, 210                               NN
·63012 DATA 255, 165, 253, 41, 15, 24, 10
 5, 65, 32, 210                             IG
·63013 DATA 255, 169, 13, 32, 210, 255, 1
 73, 141, 2, 41                             EN
·63014 DATA 1, 208, 249, 230, 63, 208, 2,
  230, 64, 230                              GJ
·63015 DATA 251, 208, 2, 230, 252, 76, 74
 , 3, 169, 236                              IK
·63016 DATA 160, 3, 32, 30, 203, 166, 63,
  165, 64, 32                               HG
·63017 DATA 205, 221, 169, 13, 32, 210, 2
 55, 96, 230, 251                           CK
·63018 DATA 208, 2, 230, 252, 96, 0, 76,
 73, 78, 69                                 JF
·63019 DATA 83, 58, 32, 0, 76, 73, 78, 69
 , 32, 35                                   OH
·63020 DATA 32, 0, 0, 0, 0, 0               LH
```

C-64 BUG REPELLENT

By Michael Kleinert and David Barron

Type in, SAVE, and RUN the *Bug Repellent*. Type NEW, then type in or LOAD the *Ahoy!* program you wish to check. When that's done, SAVE your program (don't RUN it!) and type SYS 49152 [RETURN]. You'll be asked if you want the line value codes displayed on the screen or dumped to the printer. If you select screen, it will appear there.

The table will move quickly, too quickly for most mortals to follow. To pause the listing depress and hold the SHIFT key. To pause for an extended period, depress SHIFT LOCK. As long as it is locked, the display will remain frozen.

Compare the codes your machine generates to the codes listed to the right of the respective program lines (within the white border). If you spot a difference, an error exists in that line. Jot down the numbers of lines where contradictions occur, LIST each line, spot the errors, and correct them.

```
·5000 FOR X = 49152 TO 49488 :READ Y:POKE
 X,Y:NEXT:END                               GJ
·5001 DATA 32, 161, 192, 165, 43, 133, 25
 1, 165, 44, 133                            DL
·5002 DATA 252, 160, 0, 132, 254, 32, 37,
  193, 234, 177                             DB
·5003 DATA 251, 208, 3, 76, 138, 192, 230
 , 251, 208, 2                              OF
·5004 DATA 230, 252, 76, 43, 192, 76, 73,
  78, 69, 32                                KN
·5005 DATA 35, 32, 0, 169, 35, 160, 192,
 32, 30, 171                                CA
·5006 DATA 160, 0, 177, 251, 170, 230, 25
 1, 208, 2, 230                             CE
·5007 DATA 252, 177, 251, 32, 205, 189, 1
 69, 58, 32, 210                            JE
```

```
•5008 DATA 255, 169, 0, 133, 253, 230, 25
 4, 32, 37, 193                           CL
•5009 DATA 234, 165, 253, 160, 0, 76, 13,
 193, 133, 253                            NB
•5010 DATA 177, 251, 208, 237, 165, 253,
 41, 240, 74, 74                          MB
•5011 DATA 74, 74, 24, 105, 65, 32, 210,
 255, 165, 253                            EP
•5012 DATA 41, 15, 24, 105, 65, 32, 210,
 255, 169, 13                             GH
•5013 DATA 32, 220, 192, 230, 63, 208, 2,
 230, 64, 230                             AN
•5014 DATA 251, 208, 2, 230, 252, 76, 11,
 192, 169, 153                            NG
•5015 DATA 160, 192, 32, 30, 171, 166, 63
 , 165, 64, 76                            BF
•5016 DATA 231, 192, 96, 76, 73, 78, 69,
 83, 58, 32                               EP
•5017 DATA 0, 169, 247, 160, 192, 32, 30,
 171, 169, 3                              PJ
•5018 DATA 133, 254, 32, 228, 255, 201, 8
 3, 240, 6, 201                           FK
•5019 DATA 80, 208, 245, 230, 254, 32, 21
 0, 255, 169, 4                           FL
•5020 DATA 166, 254, 160, 255, 32, 186, 2
 55, 169, 0, 133                          CL
•5021 DATA 63, 133, 64, 133, 2, 32, 189,
 255, 32, 192                             GC
•5022 DATA 255, 166, 254, 32, 201, 255, 7
 6, 73, 193, 96                           NN
•5023 DATA 32, 210, 255, 173, 141, 2, 41,
 1, 208, 249                              NH
•5024 DATA 96, 32, 205, 189, 169, 13, 32,
 210, 255, 32                             IM
•5025 DATA 204, 255, 169, 4, 76, 195, 255
 , 147, 83, 67                            KC
•5026 DATA 82, 69, 69, 78, 32, 79, 82, 32
 , 80, 82                                 DC
•5027 DATA 73, 78, 84, 69, 82, 32, 63, 32
 , 0, 76                                  ML
•5028 DATA 44, 193, 234, 177, 251, 201, 3
 2, 240, 6, 138                           GN
•5029 DATA 113, 251, 69, 254, 170, 138, 7
 6, 88, 192, 0                            JK
•5030 DATA 0, 0, 0, 230, 251, 208, 2, 230
 , 252, 96                                NA
•5031 DATA 170, 177, 251, 201, 34, 208, 6
 , 165, 2, 73                             DM
•5032 DATA 255, 133, 2, 165, 2, 208, 218,
 177, 251, 201                            JA
•5033 DATA 32, 208, 212, 198, 254, 76, 29
 , 193, 0, 169                            FM
•5034 DATA 13, 76, 210, 255, 0, 0, 0      PA
```

ULTRA-MAIL

FROM PAGE 13

START-UP

```
•1 PRINT"[CLEAR]"                              HH
•2 PRINT"[3"[DOWN]"[4"[RIGHT]"]INSERT DAT
 A DISK IN DRIVE 8"                            GK
•3 PRINT"[DOWN][3"[RIGHT]"]PRESS 'D' WHEN
  DONE"                                        IH
•4 GET X$:IFX$=""THEN4                         CE
•5 IF X$<>"D"THEN4                             LG
•6 PRINT"[DOWN][3"[RIGHT]"]FORMATING DISK
  PLEASE WAIT"                                 NB
•7 OPEN15,8,15,"N0:DATA DISK,01":CLOSE15       GK
•8 FORT=1TO1000:NEXT                           NO
•9 PRINT"[DOWN][3"[RIGHT]"]CREATING DATA
 FILE"                                         EE
•10 OPEN1,8,15                                 MP
•20 OPEN2,8,3,"0:LIST,L,"+CHR$(110)            CG
•30 GOSUB 900                                  DB
•40 PRINT#1,"P"CHR$(3)CHR$(0)CHR$(3)CHR$(
 1)                                            MG
•50 GOSUB 900                                  DB
•60 PRINT#2,0:PRINT#1,"P"CHR$(3)CHR$(0)CH
 R$(3)CHR$(1)                                  FL
•70 INPUT#2,X:PRINTX                           GG
•300 CLOSE2:CLOSE1:PRINT"[RIGHT][RIGHT]IN
 SERT PROGRAM DISK":PRINT"[RIGHT][RIGHT]L
 OAD AND RUN MAIL":END                         OA
•900 INPUT#1,E,B$,C,D:PRINTE                   MP
•910 IF(E=50) OR (E<20)THEN RETURN             ON
•920 PRINT E;B$;C;D:STOP:RETURN                CI
```

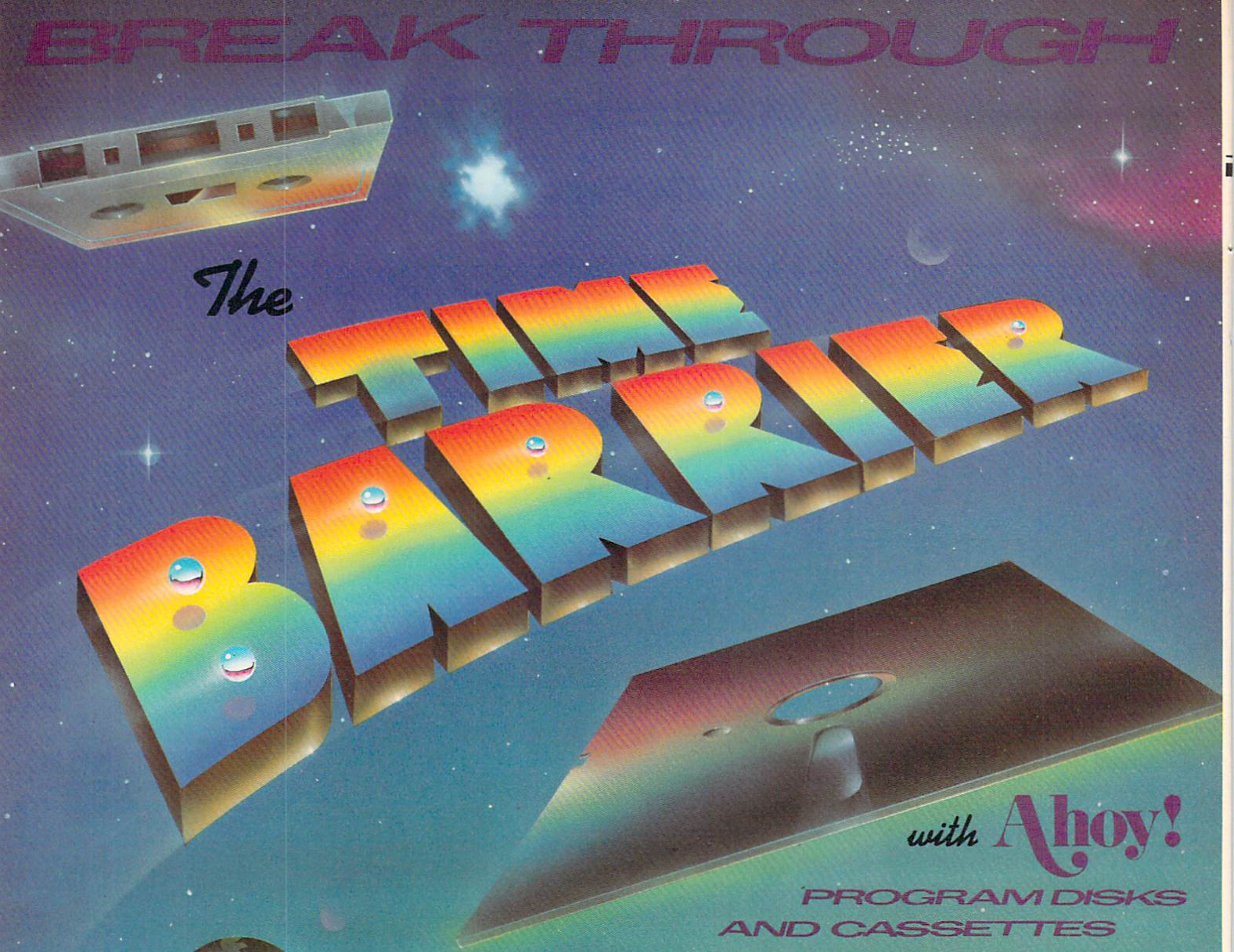
MAIN PROGRAM

```
•10 GOSUB 25000                                IG
•15 DIMV(8):V(1)=1:V(2)=14:V(3)=30:V(4)=5
 0:V(5)=70:V(6)=83:V(7)=88:V(8)=98:VJ=0        EE
•20 DIMN$(8):DIMQ(15):DIMSR$(750):DIMJ$(5
 ):DIMDL(50):DIMSQ(750):DIMSF$(750)            OO
•25 Q(0)=5:Q(1)=20:Q(2)=30:Q(3)=11:Q(4)=5
 :Q(5)=5:FL=0                                  NJ
•30 S(1)=12:S(2)=15:S(3)=19:S(4)=19:S(5)=
 12:S(6)=4:S(7)=9:S(8)=12                      MC
•31 OPEN15,8,15,"I":CLOSE15                    OH
•32 PRINT"[CLEAR]":POKE 53280,0:POKE 5328
 1,6                                           OJ
•35 FOR A=1 TO 8:PRINT:NEXT A:PRINTTAB(12
 );"[WHITE]INITIALIZING[3"."]"                 LO
•36 PRINT TAB(12);"[DOWN]PLEASE STAND BY"      IB
•37 FORY=1TO750:SQ(Y)=Y:NEXT                   FI
•40 OPEN1,8,15:OPEN2,8,3,"0:LIST":GOSUB70
 00                                            GI
•41 PRINT#1,"P"CHR$(3)CHR$(0)CHR$(3)CHR$(
 1)                                            MG
•42 INPUT#2,NN$:NN=VAL(NN$):IFNN=0THEN110      JO
•45 FOR A=1 TO 7:PRINT:NEXT A                  GJ
•50 FORI=1TONN                                 PC
•51 FORT=1TO5:NEXTT                            KL
•53 GOSUB11000:IFFL=1THENFL=0:GOTO60           GH
•60 NEXTI                                      MN
•110 PRINT"[CLEAR]":CG=0:POKE 53280,14:PO
 KE 53281,0:GOSUB 23000                        CP
•111 PRINT#1,"P"CHR$(3)CHR$(0)CHR$(2)CHR$
 (1)                                           LN
•115 PRINT TAB(12);"[c 7][s U][13"[s C]"]
 [s I]"                                        DB
•116 PRINT TAB(7);"[s U][23"[s C]"][s I]"      DB
•120 PRINT TAB(3);"[3"[s C]"]<[s -][YELLO
 W]* S E L E C T I O N S *[BLUE][s -]>[3"
 [s C]"]"                                      LP
•125 PRINT TAB(7);"[s J][23"[s C]"][s K]"      II
•126 PRINT TAB(12);"[s J][13"[s C]"][s K]
 "                                             JC
•150 PRINTTAB(11)"[DOWN][DOWN][c 7][RVSON
 ][1][RVSOFF] [YELLOW]ADD NAME"                BJ
•160 PRINTTAB(11)"[c 7][RVSON][2][RVSOFF]
 [YELLOW]CHANGE/INQUIRE"                       BD
•170 PRINTTAB(11)"[c 7][RVSON][3][RVSOFF]
 [YELLOW]PRINT COMMANDS"                       DI
•180 PRINTTAB(11)"[c 7][RVSON][4][RVSOFF]
 [YELLOW]DELETE NAME"                          AO
•185 PRINTTAB(11)"[c 7][RVSON][5][RVSOFF]
 [YELLOW]SEARCH/SCROLL"                        JD
•190 PRINTTAB(11)"[c 7][RVSON][6][RVSOFF]
 [YELLOW]SORT LIST"                            IP
•195 PRINTTAB(11)"[c 7][RVSON][7][RVSOFF]
 [YELLOW]EXIT PROGRAM"                         FC
•200 FOR A=1 TO 6:PRINT:NEXT A                 GG
•210 PRINTTAB(11)"[RED]YOUR CHOICE? [c 7]
 [RVSON][1-7]"                                 GG
•220 GETM$:IFM$=""THEN220                      ID
•230 M=VAL(M$):IFM<1ORM>7THEN220               FJ
•235 GOSUB 23050                               JJ
•240 ON M GOTO 1000,2000,10000,4000,12000
 ,14000,5000                                   KJ
•300 REM WRITE RTN.                            FA
•301 IFDF=1ANDCG=0THENSR$(Y)=N$(1):SF$(Y)
 =N$(2):GOTO305                                JA
•302 IFCG=1THEN305                             DL
•303 SR$(I)=N$(1):SF$(I)=N$(2)                 NP
•305 R1=I:R2=0:IFR1>255THENR2=INT(R1/256)
 :R1=R1-(256*R2)                               MC
•310 FORL=1TO8:PRINT#1,"P"CHR$(3)CHR$(R1)
 CHR$(R2)CHR$(V(L))                            GI
•315 IFLEN(N$(L))>S(L)THENN$(L)=LEFT$(N$(
 L),S(L))                                      AB
•320 PRINT#2,N$(L):NEXT                        MF
•325 RETURN                                    IM
•400 REM READ RTN.                             OH
•405 R1=I:R2=0:IFR1>255THENR2=INT(R1/256)
 :R1=R1-(256*R2)                               MC
•410 PRINT#1,"P"CHR$(3)CHR$(R1)CHR$(R2)CH
 R$(1)                                         BL
•415 GET#2,Z$:IFZ$="[PI]"THENFL=1:RETURN       AE
•425 FORL=1TO8:PRINT#1,"P"CHR$(3)CHR$(R1)
 CHR$(R2)CHR$(V(L))                            GI
```

```
·430 INPUT#2,N$(L):NEXT                          NB
·435 RETURN                                      IM
·1000 PRINT "[CLEAR]":POKE 53280,6:POKE 5
 3281,1                                          OI
·1010 PRINT "[BLUE][RVSON][UP][14" "][RED
 ][c A][10"[s C]"][c S][BLUE][14" "]"            FD
·1020 PRINT "[BLUE][RVSON][UP][14" "][RED
 ][s -] ADD NAME [s -][BLUE][14" "]"             LM
·1030 PRINT "[RED][RVSON][UP][14" "][c Z]
 [10"[s C]"][c X][14" "]"                        IA
·1035 PRINT "[RED][RVSON][UP][40" "]"            GA
·1036 PRINT "[RED][RVSON][UP][5" "]"             DA
·1040 PRINT "[RED][RVSON][3" "]1.[RVSOFF]
 [BLUE] REC #[s C][s C]>"                        JI
·1050 PRINT "[RED][RVSON][3" "]2.[RVSOFF]
 [BLUE] L.NAME[s C]>"                            HL
·1055 PRINT "[RED][RVSON][3" "]3.[RVSOFF]
 [BLUE] F.NAME[s C]>"                            CA
·1060 PRINT "[RED][RVSON][3" "]4.[RVSOFF]
 [BLUE] ADDRE[s C][s C]>"                        BB
·1065 PRINT "[RED][RVSON][3" "]5.[RVSOFF]
 [BLUE] ADDRE2[s C]>"                            EL
·1070 PRINT "[RED][RVSON][3" "]6.[RVSOFF]
 [BLUE] CITY[3"[s C]"]>"                         FB
·1080 PRINT "[RED][RVSON][3" "]7.[RVSOFF]
 [BLUE] STATE[s C][s C]>"                        PP
·1090 PRINT "[RED][RVSON][3" "]8.[RVSOFF]
 [BLUE] ZIP[4"[s C]"]>"                          AK
·1091 PRINT "[RED][RVSON][3" "]9.[RVSOFF]
 [BLUE] TELE[3"[s C]"]>"                         GB
·1092 PRINT "[5"[DOWN]"]"                        CM
·1093 FORA=1TO3:PRINTTAB(8);"[RED][RVSON]
 [25" "]":NEXT                                   FO
·1095 PRINT TAB(9);"[RED][RVSON][UP][UP]P
 RESS 'RETURN' FOR MENU"                         IL
·1100 GOSUB1700:IFN$(1)=""THEN110                EG
·1109 C$=""                                      GB
·1110 PRINT"[HOME]":FOR A=1TO 18:PRINT:NE
 XT:PRINTTAB(9);"[RED][RVSON][23" "]"            AO
·1112 PRINT TAB(11);"[RED][RVSON][UP]ANY
 CHANGES [Y/N]"                                  ID
·1115 GET C$:IFC$="" THEN 1115                   JG
·1116 IF C$="N" THEN 1800                        LN
·1120 IFC$="Y" THEN GOSUB 1850:GOTO 1110         HC
·1125 IF C$<>"N" OR C$<>"Y" THEN 1115            HF
·1130 GOTO1110                                   FC
·1700 FORL=1TO8:N$(L)="":NEXTL:IFDF=1THEN
 21000                                           FB
·1701 PRINT "[HOME]":PRINT TAB(14);"[BLUE
 ][3"[DOWN]"]";NN+1                              MC
·1702 N$(1)=""                                   OP
·1705 INPUT"[14"[RIGHT]"][RVSON]";N$(1)          GM
·1709 IFN$(1)=""THENRETURN                       MN
·1710 INPUT"[14"[RIGHT]"][RVSON]";N$(2):I
 FN$(2)=""THENN$(2)="NONE"                       PB
·1711 INPUT"[14"[RIGHT]"][RVSON]";N$(3):I
 FN$(3)=""THENN$(3)="NONE"                       BC
·1712 INPUT"[14"[RIGHT]"][RVSON]";N$(4):I
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
 FN$(4)=""THENN$(4)="NONE"                       CP
·1713 INPUT"[14"[RIGHT]"][RVSON]";N$(5):I
 FN$(5)=""THENN$(5)="NONE"                       EA
·1715 INPUT"[14"[RIGHT]"][RVSON]";N$(6):I
 FN$(6)=""THENN$(6)="NONE"                       MN
·1717 INPUT"[14"[RIGHT]"][RVSON]";N$(7):I
 FN$(7)=""THENN$(7)="NONE"                       CO
·1719 INPUT"[14"[RIGHT]"][RVSON]";N$(8):I
 FN$(8)=""THENN$(8)="NONE"                       AL
·1720 IFDF=0THEN1729                             GL
·1726 I=DL(VJ):FORY=1TONN:IFSQ(Y)=ITHEN17
 30                                              KH
·1727 NEXTY                                      NN
·1729 I=NN+1:NN=NN+1                             IN
·1730 RETURN                                     IM
·1800 PRINT"[CLEAR]":PRINTTAB(15)"[9"[DOW
 N]"]WORKING[7"."]":GOSUB300:IFVJ=-1THENV
 J=0                                             GL
·1810 DL(VJ)=0:VJ=VJ-1:IFVJ=0THENDF=0            GA
·1821 FORE=1TO250:NEXT:GOTO110                   DC
·1850 PRINT TAB(11);"[RED][RVSON][UP][19"
 "]"                                             FK
·1852 PRINT TAB(12);"[UP][RED][RVSON]WHIC
 H LINE [2-9]";:INPUT C:IFC<2ORC>9THENPRI
 NT:GOTO 1852                                    GI
·1855 ON C GOSUB 6000,6100,6200,6300,6400
 ,6500,6600,6700,6800                            GC
·1860 RETURN                                     IM
·2000 REM CHANGE/INQUIRE RTN.                    HA
·2010 PRINT"[CLEAR]":POKE 53280,6:POKE 53
 281,1:CG=1                                      HE
·2015 PRINT "[BLUE][RVSON][UP][11" "][RED
 ][c A][16"[s C]"][c S][BLUE][11" "]"            HP
·2020 PRINT "[BLUE][RVSON][UP][11" "][RED
 ][s -] CHANGE[s N]INQUIRE [s -][BLUE][11
 " "]"                                           PM
·2025 PRINT "[RED][RVSON][UP][11" "][RED]
 [c Z][16"[s C]"][c X][11" "]"                   BI
·2030 PRINT "[RED][RVSON][UP][40" "]"            GA
·2035 PRINT "[RED][RVSON][UP][5" "]"             DA
·2040 PRINT "[RED][RVSON][3" "]1.[RVSOFF]
 [BLUE] REC-#-->";" ";0                          NI
·2050 PRINT "[RED][RVSON][3" "]2.[RVSOFF]
 [BLUE] L.NAME[s C]>"                            HL
·2055 PRINT "[RED][RVSON][3" "]3.[RVSOFF]
 [BLUE] F.NAME[s C]>"                            CA
·2060 PRINT "[RED][RVSON][3" "]4.[RVSOFF]
 [BLUE] ADDRE[s C][s C]>"                        BB
·2065 PRINT "[RED][RVSON][3" "]5.[RVSOFF]
 [BLUE] ADDRE2[s C]>"                            EL
·2070 PRINT "[RED][RVSON][3" "]6.[RVSOFF]
 [BLUE] CITY[3"[s C]"]>"                         FB
·2080 PRINT "[RED][RVSON][3" "]7.[RVSOFF]
```


BREAK THROUGH
The TIME BARRIER
with Ahoy!
PROGRAM DISKS
AND CASSETTES

```
[BLUE] STATE[s C][s C]>"                    PP
•2090 PRINT "[RED][RVSON][3" "]8.[RVSOFF]
[BLUE] ZIP[4"[s C]"]>"                      AK
•2095 PRINT "[RED][RVSON][3" "]9.[RVSOFF]
[BLUE] TELE[3"[s C]"]>"                     GB
•2096 PRINT "[5"[DOWN]"]"                   CM
•2097 FORA=1TO3:PRINTTAB(8);"[RED][RVSON]
[25" "]":NEXT                               FO
•2098 PRINT TAB(13);"[RED][RVSON][UP][UP]
ENTER 0 FOR MENU"                           NO
•2100 INPUT"[BLUE][HOME][5"[DOWN]"[15"[RI
GHT]"][RVSON]";I:IFI<0ORI>1024THEN2100      NF
•2105 IFI=0THENCG=0:GOTO110                 EC
•2110 GOSUB400:IFFL=1THENFL=0:GOTO2100      PC
•2115 PRINT"[HOME][5"[DOWN]"]"              EP
•2120 FORL=1TO8                             JM
•2125 PRINT"[16"[RIGHT]"][RVSON]";N$(L)     PA
•2130 NEXT                                  IA
•2140 FORII=1 TO 7:PRINT:NEXTII             BG
•2210 PRINT TAB(12);"[RED][RVSON]ANY CHAN
GES ? [Y/N]"                                KA
•2212 GET C$:IF C$="" THEN 2212             JD
•2213 IF C$="N" THEN 2300                   KD
•2214 IFC$="Y"THENGOSUB1850:GOTO2210        HM
•2215 IF C$<>"Y" OR C$<>"N" THEN 2212       DK
•2220 GOTO2210                              EI
•2300 REM                                   JD
•2305 PRINT TAB(11);"[UP][UP][RED][RVSON]
PRESS 'F1' TO CONT"                         CA
•2310 PRINT TAB(11);"[RVSON]PRESS 'F3' TO
RECORD"                                     IA
•2312 PRINT TAB(11);"[RVSON]PRESS 'F7' TO
EXIT"                                       OM
•2315 GETQ$:IFQ$=""THEN2315                 NF
•2320 IFQ$="[F1]"THEN2000                   PN
•2322 IFQ$="[F7]"THENCG=0:GOTO110           LI
•2325 IFQ$="[F3]"THEN GOSUB 300:GOTO2000    NO
•2330 GOTO2315                              FE
•3000 REM PRINT RTN.                        DK
•3005 PRINT"[CLEAR]"                        HH
•3010 PRINTTAB(14)"PRINT LABELS"            CA
•3015 PRINTTAB(14)"[12"*"]"                 BB
•3020 PRINTTAB(3)"[DOWN][DOWN]ENTER 0 TO
RETURN TO RETURN TO MENU"                   DI
•3021 INPUT"[3"[RIGHT]"]ON SORT 1/YES 0/N
O";FG                                       JL
•3022 PRINT"ENTER -1 FOR ALL"               MD
•3025 INPUT"[DOWN][DOWN]STARTING REC-#";S
S                                           JF
•3026 IFSS=0THEN110                         FC
•3027 IFSS=-1THENED=NN:SS=1:GOTO3031        HL
•3030 INPUT"  ENDING REC-#";ED:IFED>NNTHE
NED=NN                                      JG
•3031 KY$="":FV$=""                         LG
•3035 INPUT"[3" "]KEYED FIELD";KY$:IFKY$=
""THEN3110                                  CN
•3040 INPUT"[3" "]FIELD VALUE";FV$          DD
•3045 GOSUB13000:IFCK=1THEN3000             DN
•3110 PRINT"[DOWN][DOWN][3"[RIGHT]"]SET T
OP OF FORM"                                 NF
•3111 PRINT"[3"[RIGHT]"]WHEN DONE ENTER '
P' TO PRINT"                                GM
•3112 PRINT"[3"[RIGHT]"]ENTER 'Q' TO QUIT
"                                           IM
•3113 PRINT"[3"[RIGHT]"]ENTER 'B' TO STOP
PRINTING"                                   IE
•3118 OPEN3,4                               DN
•3119 CMD3:PRINT#3," "                      HI
•3120 GETW$:IFW$=""THEN3120                 BA
•3125 IFW$="Q"THENCLOSE3:GOTO110            DD
•3130 IFW$<>"P"THEN3120                     IJ
•3135 FORU=SSTOED:I=U:IFFG=1THENI=SQ(U)     NM
•3136 GETW$:IFW$="B"THEN3170                FL
•3140 GOSUB400:IFFL=1THENFL=0:GOTO3165      AA
•3141 IFFV$=""THEN3145                      LG
•3142 IFN$(C)<>FV$THEN3165                  CE
•3145 PRINT#3,N$(2);" ";N$(1)               OH
•3150 PRINT#3,N$(3)                         OM
•3155 PRINT#3,N$(5);" ";N$(6);" ";N$(7)     DH
•3160 PRINT#3:PRINT#3:PRINT#3               PC
•3165 NEXT                                  IA
•3170 PRINT#3,"[3" "]":CLOSE3               KA
•3175 PRINT"[DOWN][3"[RIGHT]"][RVSON]LABE
LS PRINTED"                                 CA
•3180 PRINT"[DOWN][3"[RIGHT]"]ENTER C TO
RETURN TO MENU"                             EP
•3185 GETW$:IFW$=""THEN3185                 BB
•3190 GOTO 110                              CC
•4000 REM DELETE RTN.                       HP
•4001 PRINT"[CLEAR]":POKE 53280,6:POKE 53
281,0                                       OJ
•4002 IFVJ=50THEN110                        IB
•4005 PRINT TAB(10)"[YELLOW][s U][17"[s *
]"][s I]"                                   CD
•4006 PRINT TAB(10)"[s -] * DELETE NAME *
[s -]"                                      JF
•4010 PRINT TAB(10)"[YELLOW][s J][17"[s *
]"][s K]"                                   FK
•4011 FOR A=1 TO 17:PRINT:NEXT A            JO
•4012 PRINTTAB(12)"ENTER 0 FOR MENU"        EK
•4015 INPUT"[HOME][8"[DOWN]"[8"[RIGHT]"]E
NTER RECORD NUMBER";I                       FA
•4020 IFI<0ORI>750THEN4000                  GJ
•4025 IFI=0THEN110                          PE
•4030 GOSUB400:IFFL=1THENFL=0:GOTO4000      CP
•4035 PRINT"[YELLOW][DOWN][7"[RIGHT]"]NAM
E  ";"[RED]";N$(2)+" "+N$(1)                FH
•4040 PRINT"[YELLOW][DOWN][7"[RIGHT]"]ADD
RE ";"[RED]";N$(3)                          GO
•4045 INPUT"[YELLOW][8"[DOWN]"[9"[RIGHT]"
]CORRECT PERSON (Y-N)";Q$                   LM
•4050 IFQ$="N"THEN4000                      LA
•4055 IFQ$<>"Y"THEN4045                     IJ
•4060 INPUT"[YELLOW][UP][8"[RIGHT]"]DELET
E THIS PERSON(Y-N)";Y$                      KE
•4065 IFY$="N"THEN4000                      LI
```

```
•4070 IFY$<>"Y"THEN4060                        JM
•4071 DF=1                                     HA
•4072 VJ=VJ+1                                  NG
•4073 DL(VJ)=I                                 HL
•4075 PRINT"[UP][7"[RIGHT]"][RED][4" "][R
 VSON]PERSON DELETED[3"!"][RVSOFF][4" "]"      PC
•4076 FORY=1TONN                               AC
•4077 IFSQ(Y)=ITHENSR$(Y)="[10"*"]":SF$(Y
 )="[10"*"]":GOTO4080                          GC
•4078 NEXTY                                    NN
•4080 D$="[PI]"                                GF
•4085 R1=I:R2=0:IFR1>255THENR2=INT(R1/256
 ):R1=R1-(256*R2)                              MC
•4090 PRINT#1,"P"CHR$(3)CHR$(R1)CHR$(R2)C
 HR$(1)                                        BL
•4095 PRINT#2,D$                               GA
•4100 PRINT"[UP][9"[RIGHT]"]ENTER C TO CO
 NTINUE[6" "]"                                 IA
•4105 GETW$:IFW$=""THEN4105                    OG
•4110 IFW$<>"C"THEN4105                        FK
•4115 GOTO4000                                 FF
•5000 PRINT#1,"P"CHR$(3)CHR$(0)CHR$(3)CHR
 $(1)                                          MG
•5001 NN$=STR$(NN)                             DD
•5002 PRINT#2,NN$                              KO
•5010 CLOSE2:CLOSE1:PRINT CHR$(147):POKE
 53280,14:POKE 53281,6:END                     MO
•6000 INPUT"[HOME][5"[DOWN]"[15"[RIGHT]"]
 [RVSON]";I:IFI<0ORI>1024THEN6000              KK
•6010 RETURN                                   IM
•6100 INPUT"[HOME][6"[DOWN]"[15"[RIGHT]"]
 [RVSON]";N$(1):X=14:GOSUB 6900:RETURN         LB
•6200 INPUT"[HOME][7"[DOWN]"[15"[RIGHT]"]
 [RVSON]";N$(2):X=13:GOSUB 6900:RETURN         NE
•6300 INPUT"[HOME][8"[DOWN]"[15"[RIGHT]"]
 [RVSON]";N$(3):X=12:GOSUB 6900:RETURN         JK
•6400 INPUT"[HOME][9"[DOWN]"[15"[RIGHT]"]
 [RVSON]";N$(4):X=11:GOSUB 6900:RETURN         EP
•6500 INPUT"[HOME][10"[DOWN]"[15"[RIGHT]"
 ][RVSON]";N$(5):X=10:GOSUB 6900:RETURN        CF
•6600 INPUT"[HOME][11"[DOWN]"[15"[RIGHT]"
 ][RVSON]";N$(6):X=9:GOSUB 6900:RETURN         FO
•6700 INPUT"[HOME][12"[DOWN]"[15"[RIGHT]"
 ][RVSON]";N$(7):X=8:GOSUB 6900:RETURN         BL
•6800 INPUT"[HOME][13"[DOWN]"[15"[RIGHT]"
 ][RVSON]";N$(8):X=7:GOSUB 6900:RETURN         DD
•6900 FORII=1 TO X:PRINT:NEXTII:RETURN         BC
•7000 REM ERROR CHECK RTN.                     NA
•7005 INPUT#1,EN,EM$,ET,ES                     OA
•7010 IFEN=50THENSTOP                          HJ
•7015 IFEN=0THENRETURN                         DI
•7020 PRINT"[CLEAR]":PRINT"[9"[DOWN]"[8"[
 RIGHT]"]ERROR STATUS"                         NF
•7025 PRINT"[3"[RIGHT]"]";EN;EM$;ET,ES         GJ
•7030 PRINT"[3"[DOWN]"[9"[RIGHT]"]PLEASE
 CHECK THE ERROR"                              NG
•7035 PRINT"[8"[RIGHT]"]EXECUTION TERMANA
 TED"                                          LM
•7040 CLOSE2:CLOSE1:STOP                       KG
•8000 REM                                      JD
•8005 PRINT"[CLEAR]":CN=0                      CJ
•8010 PRINTTAB(8)"PRINT GENERAL LIST"          MN
•8015 PRINTTAB(8)"[18"*"]"                     KO
•8020 PRINTTAB(3)"[DOWN][DOWN]ENTER 0 TO
 RETURN TO RETURN TO MENU"                     DI
•8021 INPUT"[3"[RIGHT]"]ON SORT 1/YES 0/N
 O";FG                                         JL
•8022 PRINT"ENTER -1 FOR ALL"                  MD
•8024 KY$="":FV$=""                            LG
•8025 INPUT"[DOWN][DOWN]STARTING REC-#";S
 S                                             JF
•8026 IFSS=0THEN110                            FC
•8027 IFSS=-1THENED=NN:SS=1:GOTO8031           HA
•8030 INPUT"  ENDING REC-#";ED:IFED>NNTHE
 NED=NN                                        JG
•8031 INPUT"[3" "]KEYED FIELD";KY$:IFKY$=
 ""THEN8110                                    CE
•8032 INPUT"[3" "]FIELD VALUE";FV$             DD
•8045 GOSUB13000:IFCK=1THEN8000                IC
•8110 PRINT"[DOWN][DOWN][3"[RIGHT]"]SET T
 OP OF FORM"                                   NF
•8112 PRINT"[3"[RIGHT]"]WHEN DONE ENTER '
 P' TO PRINT"                                  GM
•8113 PRINT"[3"[RIGHT]"]ENTER 'Q' TO QUIT
 "                                             IM
•8114 PRINT"[3"[RIGHT]"]ENTER 'B' TO STOP
  PRINTING"                                    IE
•8118 OPEN3,4                                  DN
•8119 CMD3:PRINT#3," "                         HI
•8120 GETW$:IFW$=""THEN8120                    PB
•8125 IFW$="Q"THEN110                          JD
•8130 IFW$<>"P"THEN8120                        JK
•8132 PRINT#3,"REC# NAME:[15" "]ADDRESS:[
 22" "]CITY:";                                 NL
•8133 PRINT#3,"[6" "]ST:  ZIP:"                JO
•8134 PRINT#3:PRINT#3                          NI
•8135 FORU=SSTOED:I=U:IFFG=1THENI=SQ(U)        NM
•8136 GETW$:IFW$="B"THEN8170                   EO
•8140 GOSUB400:IFFL=1THENFL=0:GOTO8165         OB
•8141 IFFV$=""THEN8143                         LL
•8142 IFN$(C)<>FV$THEN8165                     CD
•8143 GOSUB 9000                               GG
•8144 PRINT#3,A$SPC(Q(6))B$(1)SPC(Q(7))B$
 (2)SPC(Q(8))B$(3)SPC(Q(9))B$(4);              AP
•8145 PRINT#3,SPC(Q(10))B$(5)                  MK
•8146 CN=CN+1                                  LM
•8165 NEXT                                     IA
•8166 PRINT#3,"[DOWN][DOWN][RIGHT]NUMBER
 OF PERSONS PRINTED:";CN                       HG
•8170 PRINT#3,"  ":CLOSE3                      II
•8175 PRINT"[DOWN][3"[RIGHT]"][RVSON]LABE
 LS PRINTED"                                   CA
•8180 PRINT"[DOWN][3"[RIGHT]"]ENTER C TO
 RETURN TO MENU"                               EP
•8185 GETW$:IFW$=""THEN3185                    BB
•8190 GOTO 110                                 CC
```

```
•9000 A$=STR$(I)                              IM
•9010 B$(1)=N$(1)+" "+N$(2):B$(2)=N$(3):B
 $(3)=N$(5):B$(4)=N$(6):B$(5)=N$(7)           PL
•9020 L1=LEN(A$):Q(6)=Q(0)-L1                 GM
•9030 FORX=1TO5                               JH
•9042 IFLEN(B$(X))>Q(X)THENB$(X)=LEFT$(B$
 (X),Q(X)-1)                                  CC
•9043 L1=LEN(B$(X))                           FD
•9050 Q(6+X)=Q(X)-L1:NEXT:RETURN              HA
•10000 REM                                    JD
•10005 PRINT"[CLEAR]":POKE 53280,0:POKE 5
 3281,3                                       OE
•10010 PRINT TAB(10)"[BLACK][s U][19"[s *
 ]"][s I]"                                    IK
•10012 PRINT TAB(10);"[BLACK][RVSON] * PR
 INTER OPTIONS * "                            EM
•10015 PRINT TAB(10)"[BLACK][s J][19"[s *
 ]"][s K]"                                    GH
•10016 FOR I=1 TO 5:PRINT:NEXT                BN
•10017 PRINTTAB(11)"[RVSON][0][RVSOFF] RE
 TURN TO MENU"                                KE
•10020 PRINT TAB(11);"[DOWN][RVSON][1][RV
 SOFF] MAILING LABELS"                        PK
•10030 PRINT TAB(11);"[DOWN][RVSON][2][RV
 SOFF] MAILING LIST"                          FF
•10035 FOR I=1 TO 6:PRINT:NEXT                DM
•10040 PRINTTAB(11)"YOUR CHOICE [RVSON][0
 -2]"                                         IN
•10045 GETM$:IFM$=""THEN10045                 AE
•10050 M=VAL(M$)                              IB
•10051 IFM=0THEN110                           PI
•10052 IFM<1ORM>2THEN10045                    EN
•10055 ON M GOTO 3000,8000                    BH
•11000 R1=I:R2=0:IFR1>255THENR2=INT(R1/25
 6):R1=R1-(256*R2)                            MC
•11005 PRINT#1,"P"CHR$(3)CHR$(R1)CHR$(R2)
 CHR$(1)                                      BL
•11010 GET#2,Z$:IFZ$="[PI]"THENFL=1:GOTO1
 1050                                         CM
•11012 PRINT#1,"P"CHR$(3)CHR$(R1)CHR$(R2)
 CHR$(1)                                      BL
•11015 INPUT#2,SR$(I)                         DJ
•11020 PRINT#1,"P"CHR$(3)CHR$(R1)CHR$(R2)
 CHR$(14)                                     GE
•11022 INPUT#2,SF$(I):RETURN                  AI
•11050 SR$(I)="[10"*"]":SF$(I)="[10"*"]":
 DF=1:VJ=VJ+1:DL(VJ)=I:RETURN                 PF
•12000 PRINT"[CLEAR]":POKE 53280,2:POKE 5
 3281,1                                       NM
•12005 PRINTTAB(9)"[RED][s U][19"[s *]"][
 s I]"                                        PA
•12006 PRINTTAB(9)"[s -] * SEARCH[s N]SCR
 OLL * [s -]"                                 JO
•12007 PRINTTAB(9)"[s J][19"[s *]"][s K]"
 :PRINT:PRINT                                 EI
•12008 PRINTTAB(7)"[BLUE]PRESS [RVSON]F1[
 RVSOFF] TO SLOW SCROLL"                      JP
•12009 PRINTTAB(7)"PRESS [RVSON]F2[RVSOFF
 ] TO RETURN SCROLL"                          GE
•12010 PRINTTAB(7)"PRESS [RVSON]F3[RVSOFF
 ] TO LEAVE SCROLL"                           MF
•12011 PRINTTAB(7)"PRESS [RVSON]F7[RVSOFF
 ] TO STOP SCROLL"                            JO
•12012 PRINTTAB(7)"PRESS [RVSON]F8[RVSOFF
 ] TO CONTINUE SCROLL"                        MK
•12013 PRINT:PRINT                            GJ
•12014 PRINT TAB(13)"[RVSON][0][RVSOFF] F
 OR MENU"                                     BI
•12017 PRINT TAB(13)"[RVSON][1][RVSOFF] F
 OR SEARCH"                                   IN
•12018 PRINT TAB(13)"[RVSON][2][RVSOFF] F
 OR SCROLL"                                   MN
•12019 INPUT"[DOWN][DOWN][12"[RIGHT]"][RE
 D]YOUR CHOICE";M:IFM=2THEN12025              DO
•12020 IFM=0THEN110                           PI
•12021 INPUT"[DOWN][8"[RIGHT]"][RED][RVSO
 N]LAST NAME [RVSOFF]";H$                     OC
•12022 INPUT"[8"[RIGHT]"][RVSON]FIRST NAM
 E [RVSOFF]";H1$:PRINT"[CLEAR]":GOTO22000     BD
•12025 PRINT"[CLEAR]":FOR A=1 TO 8:PRINT:
 NEXT A                                       DA
•12050 FORI=1TONN                             PC
•12051 GETA$:IFA$="[F7]"THEN12100             GL
•12052 IFM=2THEN12200                         FM
•12055 IFH$=SR$(I)THEN PRINT TAB(13);SR$(
 I);SQ(I):GOTO12070                           EF
•12058 IFM=1THEN12063                         FE
•12059 FORX=1TO75:NEXTX                       NI
•12060 GOSUB20000                             IP
•12061 IFA$="[F3]"THEN110                     KI
•12062 IFFP=1ANDM=2THENFORE=1TO400:NEXT       BL
•12063 NEXTI:IFM=2THEN12070                   GC
•12065 PRINT"[5"[DOWN]"[9"[RIGHT]"]PERSON
  NOT FOUND"                                  FI
•12070 PRINT:PRINT TAB(13);"CONTINUE Y/N"     JF
•12075 GETM$:IFM$=""THEN12075                 AF
•12080 IFM$="Y"THEN12000                      OF
•12081 IFM$<>"N"THEN12075                     IG
•12085 GOTO110                                CC
•12100 PRINT                                  JJ
•12110 GETA$:IFA$="[F8]"THEN12052             GH
•12112 IFA$="[F3]"THEN110                     KI
•12115 GOTO12110                              IM
•12200 ZJ=LEN(STR$(SQ(I))):Z1=4-ZJ:PRINTT
 AB(8)SQ(I)SPC(Z1)SR$(I);" ";SF$(I)           GN
•12201 GOTO12060                              II
•13000 REM                                    JD
•13001 CK=0                                   HH
•13015 IFKY$="CITY"THENC=5:GOTO13030          AP
•13020 IFKY$="STATE"THENC=6:GOTO13030         AF
•13025 IFKY$="ZIP"THENC=7:GOTO13030           BL
•13026 CK=1                                   HG
•13030 RETURN                                 IM
•14000 REM                                    JD
•14010 PRINT"[CLEAR]":POKE 53280,14:POKE
 53281,1                                      BO
```

```
•14015 FOR A=1 TO 6:PRINT:NEXT A          GG
•14020 PRINTTAB(13)"[BLACK][3"[DOWN]"]S O
 R T I N G"                               JA
•14030 PRINTTAB(13)"[PURPLE][13"[cEP]"]"  PN
•14040 FORY=1TONN:HY$=SR$(Y)              CP
•14041 PRINT"[HOME][DOWN][RVSON]";Y       KP
•14050 FORZZ=YTONN                        HN
•14060 IFSR$(ZZ)=<HY$THENHY$=SR$(ZZ):QQ=Z
 Z                                        DC
•14070 NEXTZZ                             DN
•14080 SR$(QQ)=SR$(Y):SR$(Y)=HY$          PH
•14081 VZ=SQ(QQ):SQ(QQ)=SQ(Y):SQ(Y)=VZ    KL
•14082 H2$=SF$(QQ):SF$(QQ)=SF$(Y):SF$(Y)=
 H2$                                      II
•14085 NEXTY                              NN
•14090 GOTO110                            CC
•20000 REM                                JD
•20010 IFA$="[F1]"THENFP=1:RETURN         FD
•20015 IFA$="[F2]"THENFP=0:RETURN         HC
•20020 RETURN                             IM
•21000 PRINT "[HOME][5"[DOWN]"]";TAB(14);
 DL(VJ):GOTO 1702                         PN
•22000 REM                                JD
•22001 Y=1:Z=NN                           CF
•22010 IFY<=ZTHENGOTO22100                NH
•22020 GOTO12065                          IP
•22100 R=INT((Y+Z)/2)                     LI
•22110 IFH$=SR$(R)ANDH1$=SF$(R)THEN22140  IA
•22120 IFH$=<SR$(R)THENZ=R-1:GOTO22010    MM
•22130 IFH$=>SR$(R)THENY=R+1:GOTO22010    OM
•22140 PRINT:PRINT:PRINT TAB(13);SF$(R);"
 ";SR$(R);SQ(R):GOTO12070                 DB
•23000 POKE SI+5,8:POKE SI+6,255:POKE SI+
 4,33                                     CA
•23010 FOR A=15 TO 1 STEP -.5:POKE SI+1,1
 6:POKE SI,195:POKESI+24,A:NEXT A         AN
•23020 POKE SI+4,16:POKE SI+1,0:POKE SI,0
 :RETURN                                  DM
•23050 POKE SI+5,8:POKE SI+6,255:POKE SI+
 4,33                                     CA
•23060 FOR A=15 TO 1 STEP -.5:POKE SI+1,2
 2:POKE SI,96:POKESI+24,A:NEXT A          MK
•23070 POKE SI+4,16:POKE SI+1,0:POKE SI,0
 :RETURN                                  DM
•25000 PRINT "[CLEAR]":POKE 53280,0:POKE
 53281,0:L=21                             JP
•25005 SI=54272:FOR SN=0 TO 24:POKE SI+SN
 ,0:NEXT:PRINT "[HOME]"                   NF
•25010 FOR A=1 TO 7:READ CH$:CH=ASC(CH$)  HL
•25020 PRINT CHR$(CH);"[3" "][RVSON] [RVS
 OFF] [RVSON] [RVSOFF] [RVSON] [RVSOFF]
 [RVSON][3" "][RVSOFF] [RVSON]  [c *][RVS
 OFF] [RVSON][sEP] [c *][RVSOFF][3" "][RV
 SON] [c *][sEP] [RVSOFF] [RVSON][sEP] [c
  *][RVSOFF] [RVSON] [RVSOFF] [RVSON] "   LA
•25030 PRINT "[3" "][RVSON] [RVSOFF] [RVS
 ON] [RVSOFF] [RVSON] [RVSOFF][3" "][RVSO
 N] [RVSOFF]  [RVSON]  [RVSOFF][sEP] [RVS
 ON] [c U] [RVSOFF][3" "][RVSON] [RVSOFF]
   [RVSON] [RVSOFF] [RVSON] [c U] [RVSOFF
 ] [RVSON] [RVSOFF] [RVSON] "             PP
•25040 PRINT "[3" "][c *][RVSON] [RVSOFF]
 [sEP] [RVSON][3" "][RVSOFF] [RVSON] [RVS
 OFF]  [RVSON] [RVSOFF][c *][RVSON][c *][
 RVSOFF] [RVSON] [RVSOFF] [RVSON] [RVSOFF
 ][3" "][RVSON] [RVSOFF]  [RVSON] [RVSOFF
 ] [RVSON] [RVSOFF] [RVSON] [RVSOFF] [RVS
 ON] [RVSOFF] [R
 VSON][3" "]"                             NO
•25045 GOSUB 25300                        IN
•25050 FOR D=1 TO 70:NEXT D:NEXT A        KE
•25055 FOR D=1 TO 100:NEXT D              OO
•25060 PRINT "[HOME]":PRINT "[3"[UP]"]":S
 N=60:FOR A=1 TO 18:GOSUB 25350           ON
•25070 PRINT "[39" "]":NEXT A:PRINT:PRINT DJ
•25080 SN=50:FOR A=1 TO L:PRINT:GOSUB 254
 00:NEXT A                                AM
•25090 CT=CT+1:IF CT=2 THEN L=18          DH
•25095 IF CT=3 THEN 25110                 LE
•25100 RESTORE:GOTO 25005                 EI
•25110 PRINT "[HOME]":FOR L=1 TO 12:PRINT
 :NEXT L:FOR D=1 TO 200:NEXT D            IF
•25120 MSG$="[40" "]DESIGNED BY[3"."]"    EM
•25130 M$=RIGHT$(MSG$,LEN(MSG$)-1):C$=M$+
 LEFT$(MSG$,1)                            DB
•25140 GOSUB 25450:PRINT "[UP][YELLOW]";L
 EFT$(C$,40);:MSG$=C$                     FP
•25150 FOR D=1 TO 150:NEXT D:MSG=MSG+1:IF
  MSG=26 THEN 25170                       PK
•25160 GOTO 25130                         IO
•25170 P=55896:C=1624:FOR A=5 TO 35:CL=IN
 T(RND(1)*8)+2:POKE P+A,CL:NEXT           PK
•25180 FOR A=1 TO 16:READ N:B=B+1:POKE C+
 11+B,N:IF N<>32 THEN GOSUB 25500         MB
•25185 FOR D=1 TO 75:NEXT D:NEXT A        ON
•25190 FOR D=1 TO 500:NEXT D:PRINT "[4"[D
 OWN]"][YELLOW]";TAB(19)"AND":GOSUB 25600 HG
•25195 FOR D=1 TO 400:NEXT D              AL
•25200 P=56056:C=1784:FOR A=5 TO 35:CL=IN
 T(RND(1)*8)+2:POKE P+A,CL:NEXT           AK
•25210 B=0:FOR A=1 TO 18:READ N:B=B+1:POK
 E C+11+B,N:IF N<>32 THEN GOSUB 25500     BF
•25215 FOR D=1 TO 75:NEXT D:NEXT A        ON
•25220 FOR D=1 TO 400:NEXT D              AL
•25225 PRINT "[4"[DOWN]"[3" "]INSERT DATA
  DISK[3"."]PRESS F1 TO START"            BP
•25230 GET A$:IF A$="[F1]" THEN 25250     GK
•25240 CL=INT(RND(1)*9)+2:POKE 53280,CL:F
 OR D=1 TO 300:NEXT D:GOTO 25230          GO
•25250 RETURN                             IM
•25300 POKE SI+5,143:POKE SI+6,255:POKE S
 I+24,15:POKE SI+4,33                     KP
•25310 FORSN=50 TO 100 STEP 5:POKESI,SN-4
 0:POKESI+1,SN:NEXT SN                    FO
•25320 POKE SI+4,16:POKE SI,0:POKE SI+1,0
 :RETURN                                  FM
```

```
•25350 POKE SI+5,143:POKE SI+6,255:POKE S
 I+24,15:POKE SI+4,33                      KP
•25360 SN=SN-3:POKE SI,SN:POKE SI+1,SN+10
 0                                         PD
•25370 POKE SI+4,16:POKE SI,0:POKE SI+1,0
 :RETURN                                   FM
•25400 POKE SI+5,143:POKE SI+6,255:POKE S
 I+24,15:POKE SI+4,33:POKE SI,30           KF
•25410 SN=SN+5:POKE SI+1,SN                OF
•25420 POKE SI+4,16:POKE SI,0:POKE SI+1,0
 :RETURN                                   FM
•25450 POKE SI+5,143:POKE SI+6,255:POKE S
 I+24,15:POKE SI+4,129                     MI
•25460 POKE SI,3:POKE SI+1,35:FOR D=1 TO
 25:NEXT D                                 DF
•25470 POKE SI+4,16:POKE SI,0:POKE SI+1,0
 :RETURN                                   FM
•25500 POKE SI+5,143:POKE SI+6,255:POKE S
 I+24,15:POKE SI+4,17                      LB
•25510 SN=INT(RND(1)*100)+25:POKE SI,SN:P
 OKE SI+1,SN                               PH
•25520 POKE SI+4,16:POKE SI,0:POKE SI+1,0
 :RETURN                                   FM
•25600 POKE SI+5,143:POKE SI+6,255:POKE S
 I+24,15:POKE SI+4,129                     MI
•25610 POKE SI,2:POKE SI+1,25:FOR D=1 TO
 35:NEXT D                                 EG
•25620 POKE SI+4,16:POKE SI,0:POKE SI+1,0
 :RETURN                                   FM
•26000 DATA "[CYAN]","[c 7]","[BLUE]","[P
 URPLE]","[RED]","[c 3]","[YELLOW]"        HA
•26010 DATA 10,32,15,32,5,32,32,6,32,21,3
 2,19,32,3,32,15                           NF
•26020 DATA 2,32,15,32,2,32,32,12,32,12,3
 2,15,32,18,32,5,32,20                     NN
```

SPRITE BASICS

FROM PAGE 92

SPRITE MAKER

```
•1 REM =-= SPRITE MAKER =-=                DE
•10 INPUT"STORE IN WHICH MEMORY BLOCK (19
 2..199)";BK                               KI
•20 MEM=64*BK                              DC
•30 FOR ROW=0 TO 20                        FH
•40 READ A$                                OJ
•50 FOR BYTE=0 TO 2 : TTL=0                IC
•60 FOR BIT=7 TO 0 STEP -1                 LJ
•70 IF MID$(A$,8+8*BYTE-BIT,1)="."THEN 90  JF
•80 TTL=TTL+2[UPARROW]BIT                  OP
•90 NEXT BIT                               GG
•100 PRINT TTL,                            MB
•110 POKE MEM,TTL : MEM=MEM+1              ON
•120 NEXT BYTE                             LK
•130 PRINT                                 JJ
•140 NEXT ROW                              HN
•1001 DATA........................         HP
•1002 DATA...................**...         OP
•1003 DATA...................***..         JD
•1004 DATA....................**..         HP
•1005 DATA........................         HP
•1006 DATA...................*...*         NP
•1007 DATA....................*...         HL
•1008 DATA.................*..*...         MH
•1009 DATA..................***...         HL
•1010 DATA..............**....*...         GD
•1011 DATA.............*..*...*...         HL
•1012 DATA............*....*..*...         CD
•1013 DATA....*........*..*..**...         HL
•1014 DATA***..*...**...**........         HH
•1015 DATA****..*.*..*............         ID
•1016 DATA*..**..*....*...........         HD
•1017 DATA*...**..*...............         FH
•1018 DATA.*.**.*..*..*...........         IP
•1019 DATA..**...**..*............         FL
•1020 DATA...*....***.............         FH
•1021 DATA....*...................         HL
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

SPRITE SAVER

```
•1 REM =-= SPRITE SAVER =-=                AB
•2 REM * ADD THESE LINES            *      PJ
•3 REM * TO THE PREVIOUS PROGRAM *         CJ
•4 REM <><><><><><><><><><><><><><         BE
•5 INPUT"WHAT IS THE FILENAME FOR THIS SP
RITE";FL$                                  MJ
•25 OPEN 8,8,8,"0:"+FL$+",S,W"             DP
•35 REM >FOR CASSETTES, REPLACE  LINE 25
 WITH THIS: OPEN 8,1,1,FL$                 NG
•105 PRINT#8,TTL                           AA
•145 CLOSE 8                               NI
```

SPRITE LOADER

```
•1 REM --- SPRITE LOADER ---               DK
•10 INPUT"LOAD WHICH SPRITE FILE";FL$      IP
•20 INPUT"WHICH MEMORY BLOCK (192..199)";
 BK                                        KI
•30 MEM=64*BK                              DC
•40 OPEN 8,8,8,"0:"+FL$+",S,R"             DO
•45 REM >FOR CASSETTE, REPLACE   LINE 40
 WITH THIS: OPEN 8,1,0,FL$                 CJ
•50 FOR N=1 TO 63                          MI
•60 INPUT#8,TTL                            NM
•70 PRINT TTL,                             MB
•80 POKE MEM,TTL : MEM=MEM+1               ON
•90 NEXT N                                 NA
•100 CLOSE 8                               NI
```

SPRITE CONTROLLER

```
•1 REM =-=SPRITE CONTROLLER=-=             JB
•2 REM                                     JD
•5 REM SPRITE DATA MUST ALREADY            MD
```

```
•6 REM BE IN MEMORY.                      MJ
•9 REM                                    JD
•10 REM <>                                AM
•11 REM <>  FNSB(N) AND FNRB(N)           FA
•12 REM <>  ARE USED TO SET/RESET         GF
•13 REM <>  BIT N OF LOCATION M           HL
•14 REM <>  WHEN USED WITH A POKE         HF
•15 REM <>  AS FOLLOWS:                   AH
•16 REM <>     POKE M, FNSB(N)            JE
•17 REM <>  OR                            KM
•18 REM <>     POKE M, FNRB(N)            JB
•19 REM <>                                AM
•20 DEF FNSB(N)=PEEK(M) OR 2[UPARROW]N    HC
•30 DEF FNRB(N)=PEEK(M) AND (255-2[UPARRO
W]N)                                      ND
•35 V=53248 :REM VIC CHIP ADDRESS         JE
•40 INPUT"WHICH SPRITE # (0..7)";SN       KI
•45 IF SN<0 OR SN>7 THEN 40               FL
•50 INPUT"WHICH MEMORY BLOCK (192..199)";
BK                                        KI
•55 IF BK<192 OR BK>199 THEN 50           BG
•60 GOSUB 500 :REM * SET POINTER          CI
•70 INPUT"WHAT X COORDINATE (0..511)";X   DC
•75 IF X<0 OR X>511 THEN 70               BC
•80 GOSUB 600 :REM * SET X COORD.         BD
•90 INPUT"WHAT Y COORDINATE (0..255)";Y   LD
•95 IF Y<0 OR Y>255 THEN 90               BB
•100 GOSUB 700 :REM * SET Y COORD.        PJ
•110 INPUT"WHAT COLOR (0..15)";C          LM
•115 IF C<0 OR C>15 THEN 110              PO
•120 GOSUB 800 :REM * SET COLOR           IE
•130 PRINT"PRESS A KEY TO TURN SPRITE ON" ML
•140 GET A$:IF A$="" THEN 140             HI
•150 GOSUB 900 :REM * TURN IT ON          NN
•160 PRINT"PRESS A KEY TO TURN SPRITE OFF
"                                         PK
•170 GET A$:IF A$="" THEN 170             HH
•180 GOSUB 1000 :REM * TURN IT OFF        OI
•190 PRINT"PRESS ANY KEY"                 GG
•200 GET A$:IF A$="" THEN 200             IB
•210 GOTO 2000 :REM * MOVE SPRITE         EK
•495 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•496 REM= SET SPRITE POINTER  =           PA
•497 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•500 POKE 2040+SN,BK                      EI
•510 RETURN                               IM
•595 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•596 REM=  SET X COORDINATE   =           DJ
•597 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•600 M=V+16 : X0=X                        BJ
•610 IF X<256 THEN POKE M,FNRB(SN):GOTO 6
30                                        KP
•620 X0=X-256 : POKE M,FNSB(SN)           IB
•630 POKE V+SN*2,X0                       NO
•640 RETURN                               IM
•695 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•696 REM=  SET Y COORDINATE   =           EC
•697 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•700 POKE V+SN*2+1,Y                      IP
•710 RETURN                               IM
•795 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•796 REM=     SET COLOR       =           HJ
•797 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•800 POKE V+39+SN,C                       NN
•810 RETURN                               IM
•895 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•896 REM=   TURN SPRITE ON    =           NJ
•897 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•900 M=V+21                               FM
•910 POKE M,FNSB(SN)                      DI
•920 RETURN                               IM
•995 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•996 REM=   TURN SPRITE OFF   =           PH
•997 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=     LD
•1000 M=V+21                              FM
•1010 POKE M,FNRB(SN)                     DL
•1020 RETURN                              IM
•1994 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=    LD
•1995 REM=      MOVE SPRITE      =        DJ
•1996 REM=-=-=-=-=-=-=-=-=-=-=-=-=-=-=    LD
•2000 X=24:Y=150:DX=4:DY=2                BH
•2010 GOSUB 900                           DB
•2020 X=X+DX : Y=Y+DY                     HP
•2030 IF X>320 THEN DX=-DX                HP
•2040 IF X<24 THEN DX=-DX                 GD
•2050 IF Y>229 THEN DY=-DY                GA
•2060 IF Y<50 THEN DY=-DY                 CL
•2070 GOSUB 600 : GOSUB 700               IJ
•2080 GOTO 2020                           FF
•2090 NEXT                                IA
```

BRINGING THE SCREEN TO LIFE

FROM PAGE 90

RUNNER

```
•1 REM ANIMATE A RUNNER                           LI
•2 REM                                            JD
•3 REM PRESS SHIFT, COMMODORE, OR CTRL TO
 QUIT                                             MH
•4 REM                                            JD
•10 DIM CB(2),CM(2),RU$(2),BL$(2)                 OK
•17 REM                                           JD
•18 REM GO SET UP MEMORY AND VARIABLES            CH
•19 REM                                           JD
•20 GOSUB 400                                     CE
•95 REM                                           JD
•96 REM MAIN MOVEMENT LOOP                        EH
•97 REM                                           JD
•98 REM IF SHIFT, COMMODORE, OR CTRL ARE
PRESSED, JUMP TO ENDING ROUTINE                   HO
•99 REM                                           JD
•100 FOR I=0 TO 38:IF PEEK(653)<>0 THEN 2
00                                                IM
•107 REM                                          JD
```

```
•108 REM PRINT LEFT-FACING AND RIGHT-FACI
 NG RUNNERS IN NEW POSITION              JC
•109 REM                                 JD
•110 PRINT "[HOME][5"[DOWN]"]"TAB(39-I)BL
 $(0)RU$(0)"[HOME][7"[DOWN]"]"TAB(I)BL$(2
 )RU$(2);                                GN
•117 REM                                 JD
•118 REM CYCLE THROUGH THE CHARACTER SETS MP
•119 REM                                 JD
•120 AM=AM+1:IF AM>2 THEN AM=0           NK
•127 REM                                 JD
•128 REM SWITCH CHARACTER SET AND WAIT A
 MOMENT                                  GD
•129 REM                                 JD
•130 POKE 53272,CM(AM):FOR K=0 TO 59:NEXT
 :NEXT                                   MD
•190 PRINT "[CLEAR]":GOTO 100            MI
•197 REM                                 JD
•198 REM RESTORE VIDEO MEMORY TO NORMAL,
 RE-ENABLE INTERRUPTS,AND END            DC
•199 REM                                 JD
•200 POKE 808,237:POKE 648,4:POKE 53272,X
 X:POKE 56578,PEEK(56578) OR 3           JG
•210 POKE 56576,(PEEK(56576)AND 252)OR 3:
 END                                     GC
•397 REM                                 JD
•398 REM SET UP VIDEO MEMORY             FO
•399 REM                                 JD
•400 VM=1:SM=15:FOR I=0 TO 2:CM(I)=I*2+8:
 NEXT:XX=PEEK(53272)                     KE
•407 REM                                 JD
•408 REM CONVERT VIDEO POINTERS TO MEMORY
  ADDRESSES                              AL
•409 REM                                 JD
•410 VB=16*1024*VM:SB=VB+1024*SM:FOR I=0
 TO 2:CB(I)=CM(I)*1024+VB:NEXT           LM
•417 REM                                 JD
•418 REM LOOPS TO LOAD CHARACTER DATA    GO
•419 REM                                 JD
•420 FOR I=0 TO 16 STEP 16:FOR J=0 TO 2:B
 =CB(J)+I:FOR K=0 TO 15                  DK
•427 REM                                 JD
•428 REM READ AND POKE DATA              NE
•429 REM                                 JD
•430 READ A:POKE B+K,A:NEXT:NEXT:NEXT    HE
•437 REM                                 JD
•438 REM SET CHARACTER AND SCREEN MEMORY DB
•439 REM                                 JD
•440 SK=SM*16:POKE 648,SB/256            OK
•445 FOR I=0 TO 2:CM(I)=SK+CM(I):NEXT:POK
 E 53272,CM(0)                           CM
•447 REM                                 JD
•448 REM SET VIDEO BLOCK                 IJ
•449 REM                                 JD
•450 POKE 56578,PEEK(56578)OR 3:POKE 5657
 6,(PEEK(56576) AND 252) OR (3-VM)       FO
•457 REM                                 JD
•458 REM DISABLE SHIFT-COMMODORE AND KEYB
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
 OARD INTERRUPTS                         IE
•459 REM                                 JD
•460 PRINT CHR$(8):POKE 808,234          BL
•467 REM                                 JD
•468 REM DEFINE RUNNER STRINGS           FP
•469 REM                                 JD
•470 RU$(0)="@[DOWN][LEFT]A":RU$(2)="B[DO
 WN][LEFT]C":BL$(0)="[RIGHT] [DOWN][LEFT]
  [UP][LEFT][LEFT]":BL$(2)=" [DOWN][LEFT]
  [UP]"                                  CF
•477 REM                                 JD
•478 REM POKE IN SPACE CHARACTERS        AL
•479 REM                                 JD
•480 FOR I=0 TO 2:FOR J=32*8 TO 32*8+7:PO
 KE CB(I)+J,0:NEXT:NEXT                  AB
•490 PRINT "[CLEAR]":RETURN              DG
•997 REM                                 JD
•998 REM CHARACTER DATA                  EG
•999 REM                                 JD
•1000 DATA 24,56,24,12,78,45,30,12       LC
•1010 DATA 28,60,104,72,68,66,193,2      II
•1020 DATA 24,56,24,12,12,14,62,28       OG
•1030 DATA 44,12,28,48,24,20,24,48       PH
•1040 DATA 24,56,24,12,12,14,125,13      GH
•1050 DATA 13,28,52,36,36,36,100,12      JD
•1060 DATA 24,28,24,48,114,180,120,48    AK
•1070 DATA 56,60,22,18,34,66,131,64      EL
•1080 DATA 24,28,24,48,48,112,124,56     MN
•1090 DATA 52,48,56,12,24,40,24,12       BI
•1100 DATA 24,28,24,48,48,112,190,176    PC
•1110 DATA 176,56,44,36,36,36,38,48      PC
```

MOVERS

```
•10 DIM CB(3),CM(3),MV$(6)               FG
•20 GOSUB 400:GOSUB 300                  HP
•107 REM                                 JD
•108 REM MAIN LOOP                       PK
•109 REM                                 JD
•110 AM=AM+1:IF AM>3 THEN AM=0           NP
•117 REM                                 JD
•118 REM SWITCH CHARACTER SETS           EE
•119 REM                                 JD
•120 POKE 53272,CM(AM):FOR K=0 TO 59:NEXT PC
•127 REM                                 JD
•128 REM IS SHIFT, COMMODORE, OR CTRL PRE
 SSED?                                   CJ
•129 REM                                 JD
•130 IF PEEK(653)>0 THEN 200             HO
•190 GOTO 110                            CC
•197 REM                                 JD
•198 REM ENDING ROUTINE                  HC
•199 REM                                 JD
•200 POKE 808,237:POKE 648,4:POKE 53272,X
```

```
 X:POKE 56578,PEEK(56578)OR 3                JG
•210 POKE 56576,(PEEK(56576)AND 252)OR 3:
 PRINT "[c 7]":END                           BF
•298 REM SET UP SCREEN RANDOMLY              LB
•300 MV$(0)="[PURPLE]A":MV$(1)="[WHITE]B"
 :MV$(2)="[GREEN]C":MV$(3)="[BLACK]D":MV$
 (4)="[c 1]E":MV$(5)="[c 2]F"                NC
•310 MV$(6)="[BLACK]GH"                      FG
•340 FOR I=0 TO 35:D=INT(RND(9)*24):R=2*I
 NT(RND(9)*19):CC=INT(RND(9)*7)              AJ
•350 PRINT "[HOME]";:FOR J=0 TO D:PRINT "
 [DOWN]";:NEXT:PRINT TAB(R)MV$(CC)" ";:NE
 XT:RETURN                                   HH
•396 REM VIDEO SET-UP ROUTINES               FM
•398 REM SET VIDEO POINTERS                  IC
•400 VM=1:SM=6:FOR I=0 TO 3:CM(I)=I*2+8:N
 EXT:XX=PEEK(53272)                          OA
•408 REM SET STARTING ADDRESSES              JB
•410 VB=16*1024*VM:SB=VB+1024*SM:FOR I=0
 TO 3:CB(I)=CM(I)*1024+VB:NEXT               HJ
•418 REM POKE CHARACTER DATA INTO PLACE      LB
•420 FOR I=0 TO 3:FOR J=CB(I)+8 TO CB(I)+
 71:READ A:POKE J,A:NEXT:NEXT                CM
•428 REM SET UP BLANK CHARACTER              HK
•430 FOR I=0 TO 3:FOR J=CB(I)+32*8 TO CB(
 I)+32*8+7:POKE J,0:NEXT:NEXT                FO
•438 REM POKE NEW CHARACTER SET AND SCREE
 N MEMORY LOCATIONS                          EC
•440 SK=SM*16:POKE 648,SB/256                OK
•445 FOR I=0 TO 3:CM(I)=CM(I)+SK:NEXT:POK
 E 53272,CM(0)                               BJ
•448 REM POKE NEW VIDEO BLOCK LOCATION       IF
•450 POKE 56578,PEEK(56578)OR 3:POKE 5657
 6,(PEEK(56576)AND 252)OR (3-VM)             FO
•458 REM DISABLE SHIFT-COMMODORE AND RUN/
 STOP-RESTORE                                AJ
•460 PRINT CHR$(8):POKE 808,234              BL
•490 PRINT "[CLEAR]":RETURN                  DG
•998 REM CHARACTER DATA                      EG
•1000 DATA 0,0,0,24,24,0,0,0                 JL
•1010 DATA 124,254,146,238,254,124,0,0       JO
•1020 DATA 28,62,62,28,0,0,0,0               BC
•1030 DATA 56,64,128,128,128,128,192,192     KO
•1040 DATA 0,0,0,0,0,0,16,254                BA
•1050 DATA 0,64,64,255,255,2,2,0             HP
•1060 DATA 0,10,10,10,22,45,83,145           EH
•1070 DATA 0,80,80,80,104,172,202,137        CK
•1080 DATA 0,0,24,44,52,24,0,0               CL
•1090 DATA 124,254,146,238,254,84,124,0      OJ
•1100 DATA 0,0,0,28,62,62,28,0               PO
•1110 DATA 0,0,0,128,128,128,67,63           AD
•1120 DATA 0,0,0,0,16,56,124,254             EB
•1130 DATA 2,7,14,158,121,112,224,64         IJ
•1140 DATA 0,20,20,42,45,75,81,144           CJ
•1150 DATA 0,40,40,84,180,210,138,9          JO
•1160 DATA 0,60,122,102,102,94,60,0          BH
•1170 DATA 124,254,146,238,254,84,40,124     MB
•1180 DATA 0,0,0,0,0,28,62,127               HE
•1190 DATA 3,3,1,1,1,1,2,28                  LD
•1200 DATA 0,0,16,16,56,56,124,254           NM
•1210 DATA 24,30,24,24,24,24,120,24          MN
•1220 DATA 20,42,45,43,81,80,80,144          KJ
•1230 DATA 40,84,180,212,138,10,10,9         GE
•1240 DATA 126,191,195,203,211,195,253,12
 6                                           AA
•1250 DATA 124,254,146,238,254,84,124,0      OJ
•1260 DATA 0,0,0,28,62,62,28,0               PO
•1270 DATA 252,194,1,01,1,0,0,0              HP
•1280 DATA 16,16,56,56,124,124,254,254       GC
•1290 DATA 72,240,112,56,28,14,15,18         IJ
•1300 DATA 0,20,20,42,45,75,81,144           CJ
•1310 DATA 0,40,40,84,180,210,138,9          JO
```

Alice in Adventureland

FROM PAGE 90

```
•5 REM ALICE                                 OJ
•10 POKE53280,0:POKE53281,0:PRINT"[CLEAR]
 [c 1]":FORI=1TO7:PRINT:NEXT                 PH
•15 PRINTTAB(15)"[s U][s I][c A] [c R][s
 U][s I][c A][c S]":PRINTTAB(15)"[c Q][c
 W][s -] [s -][s -] [c Q]":PRINTTAB(15)"[
 c X][c Z][c Z][c X][c E][s J][s K][c Z][
 c X]"                                       PM
•17 PRINT:PRINTTAB(15)"BY SPIRKO":SE=5427
 2:POKESE+24,15                              MP
•20 DIMR$(5,5,5),PX(12),PY(12),SP(12),K$(
 12),A(15),B(15),C(15)                       OH
•30 DEFFNR(R)=S(0)+S(1)+S(2)+S(3)+S(4)+S(
 5)+S(6)+S(7):00=INT(14*RND(0))+1            CG
•35 FORI=1TO6:SP(I+6)=2+2*I:NEXT             GF
•40 GOSUB1700:GOSUB2000:DEFFNX(R)=R(3)+R(
 4)+R(6)+R(7)                                KG
•47 FORI=0TO5:READL$(I):NEXT:FORI=1TO4:RE
 ADDR$(I):NEXT                               JB
•48 FORI=1TO4:READQ$(I):NEXT                 CC
•50 G1=1472:G2=1730                          AF
•55 FORI=1TO12:READPX(I),PY(I):NEXT:FORI=
 7TO12:READK$(I):NEXT                        AJ
•60 FORI=1TO15:READA(I),B(I),C(I):NEXT:FO
 RI=1TO3:READJ,ST$(J):NEXT                   HE
•61 FORZ=0TO5:FORY=0TO5:FORX=0TO5:READR$(
 X,Y,Z):L=ASC(LEFT$(R$(X,Y,Z),1))            FC
•62 IFL>77ANDL<84THENGOSUB1500               DD
•64 NEXTX,Y,Z:X=4:Y=5:Z=0                    HH
•65 PRINT"[CLEAR][c 8]":PRINT:PRINT:PRINT
 "IS THIS A[c M][RVSON]N[RVSOFF]EW GAME O
 R A[c M][RVSON]C[RVSOFF]ONTINUATION?"       PM
•66 GETA$:IFA$=""THEN66                      GH
•67 IFA$="C"THEN700                          HG
•69 PRINT"[CLEAR]":PRINT:PRINT:PRINTTAB(1
 2)"[CYAN]RIGHT  [c N][RVSON]>[RVSOFF][c
 H]"                                         AG
•70 PRINTTAB(12)"LEFT[3" "][c M][RVSON]<[
```

```
RVSOFF][c G]":PRINTTAB(12)"UP[5" "][c M]
[RVSON]X[RVSOFF][c G]"                  HL
•71 PRINTTAB(12)"DOWN[3" "][c N][RVSON]Z[
RVSOFF][c G]":PRINTTAB(12)"QUIT[3" "][c
M][RVSON]F1[RVSOFF][c G]"               JM
•72 PRINTTAB(12)"YES[4" "][c N][RVSON]SPA
CE[RVSOFF][c H]":PRINTTAB(12)"NO[6" "]AN
Y KEY"                                  HP
•80 PRINT:PRINT:PRINT:PRINT:PRINTTAB(11)"
[c 8][c N][RVSON]F1[RVSOFF][c H] SLOW"  KP
•81 PRINTTAB(11)"[c N][RVSON]F3[RVSOFF][c
 G] MEDIUM":PRINTTAB(11)"[c N][RVSON]F5[
RVSOFF][c G] FAST"                      OA
•85 PRINT:PRINTTAB(12)"[c 8]WHAT SPEED? [
c 3]";                                  II
•86 PRINT"[s V][LEFT]";:GETA$:PRINT"[s +]
[LEFT]";:IFA$=""THEN86                  PF
•87 IFA$="[F1]"THENSP=4:GOTO90          OG
•88 IFA$="[F3]"THENSP=5:GOTO90          LC
•89 SP=7                                IN
•90 FORH=1TO4:READL:FORI=LTOL+62:READJ:PO
KEI,J:NEXTI,H                           OM
•100 PRINT"[CLEAR][c 1][DOWN][DOWN][RIGHT
][RIGHT][RVSON][c E][s +][s +][c R][c E]
[c W][RVSOFF][4" "][RVSON][c Q][c E][s +
][c R][c E][s +]"                       HM
•101 PRINT"[RVSON][RIGHT][RIGHT][s +][DOW
N][LEFT][c R][DOWN][LEFT][c Q][DOWN][LEF
T][s +][DOWN][LEFT][c E][DOWN][LEFT][c W
][DOWN][LEFT][s +][DOWN][LEFT][c Q][DOWN
][LEFT][s -][DOWN][LEFT][c R][DOWN][LEFT
][s +][DOWN][LEFT][c E][DOWN][LEFT][c Q]
[DOWN][LEFT][s
+][DOWN][LEFT][c Z]";                   OO
•102 PRINT"[RVSON][LEFT][c W][s +][c E][s
 +][c W][c W][c E][s *][c E][c R][c Q][c
 R][c E][s +][c X]";                    NG
•103 PRINT"[s +][UP][LEFT][c R][UP][LEFT]
[s +][UP][LEFT][c Q][UP][LEFT][c E][UP][
LEFT][c R][UP][LEFT][c W][UP][LEFT][s +]
[UP][LEFT][s -][UP][LEFT][c Q][UP][LEFT]
[c E][UP][LEFT][c Q][UP][LEFT][s +][UP][
LEFT][c E][UP][LEFT][s +][UP][LEFT]"    DB
•105 V=53248:V2=V+2:V3=V+3:VC=V+31:POKEV+
37,1:POKEV+38,10                        KK
•115 S(1)=2:POKE2041,13:POKEV+28,3:POKEV+
40,14                                   OG
•116 POKEV+2,PX(5):POKEV+3,PY(5):FORI=1TO
1500:NEXT                               GJ
•120 POKEV+41,10:POKE2042,14:POKEV+42,5:P
OKE2043,14                              GM
•130 POKEV+43,4:POKE2044,14:POKEV+44,14:P
OKE2045,14                              KB
•135 POKEV+45,7:POKE2046,14:POKE2047,14:P
OKEV+46,1                               BI
•217 LP$="[c 8][HOME][DOWN][34"[RIGHT]"]" GE
•220 PRINT"[HOME][5"[DOWN]"][YELLOW]"TAB(
26)"MAP[4" "]LEVEL"                     DI
•222 QP$="[HOME][DOWN][26"[RIGHT]"][WHITE
]"                                      CM
•224 PRINTQP$"[GREEN][UP][3"[c @]"][DOWN]
[4"[LEFT]"][c N][3"[RIGHT]"][c H][DOWN][
5"[LEFT]"][c N][3"[RIGHT]"][c H][DOWN][5
"[LEFT]"][c M][3"[RIGHT]"][c G][DOWN][4"
[LEFT]"[4"[c T]"]"                      DE
•225 SP$="[HOME][22"[DOWN]"][CYAN]"     GC
•226 PRINT"[HOME][8"[DOWN]"]"TAB(26);:GOS
UB230                                   FM
•228 PRINT"[HOME][12"[DOWN]"]"TAB(26);:GO
SUB230:GOTO250                          OP
•230 PRINT"[c 5][c A][s *][s *][c S][c A]
[s *][s *][c S][c A][s *][s *][c S][DOWN
][12"[LEFT]"][s -]  [s -][s -]  [s -][s
-]  [s -][DOWN][12"[LEFT]"][s -]  [s -][
s -]  [s -][s -]  [s -]";               EP
•232 PRINT"[DOWN][12"[LEFT]"][c Z][s *][s
 *][c X][c Z][s *][s *][c X][c Z][s *][s
 *][c X]":RETURN                        PJ
•250 PRINT"[DOWN][YELLOW]"TAB(22)"GOLD":L
C=54272                                 HB
•260 IFGS>0THENFORI=1TOGS:POKEG2+I,81:POK
EG2+I+LC,7:NEXT                         NI
•300 ER$="[38" "]":ER$=SP$+ER$+ER$      MK
•440 AX=PX(5):AY=PY(5):POKEV+21,FNR(R):PO
KEV+16,FNX(R)                           NG
•450 FORE=7TO12:IFS(E-5)>0THENPOKEV+SP(E)
,PX(E):POKEV+SP(E)+1,PY(E)              HK
•460 NEXT                               IA
•500 PRINTER$:GOSUB1100:GOSUB1200:IFSK=0T
HENGOSUB600                             EJ
•502 IFSK=1THEN510                      EF
•505 GOSUB1000                          FO
•510 SK=0:PRINTER$:S(1)=0:S(0)=0:POKEV+21
,FNR(R):GOSUB1400                       EL
•515 GOSUB1300:IFND=0THEN500            PP
•520 K=ND-(ND<3)*2+(ND>2)*2:AX=PX(K):AY=P
Y(K)                                    NM
•530 AX=PX(K):AY=PY(K):Y=Y+(ND=1)-(ND=3):
X=X-(ND=2)+(ND=4):GOTO500               OA
•600 S(1)=2:POKEV2,AX:POKEV3,AY:POKEV+21,
FNR(R)                                  PG
•610 FORI=1TO24STEP2:NX=I*(ND=4)-I*(ND=2)
:NY=I*(ND=1)-I*(ND=3)                   GO
•620 POKEV2,AX+NX:POKEV3,AY+NY:NEXT     ED
•625 AX=NX+AX:AY=AY+NY:RETURN           GA
•700 PRINT:PRINT:PRINT"ARE YOU USING [RVS
ON]T[RVSOFF]APE OR [RVSON]D[RVSOFF]ISK?" EC
•702 GETB1$:ON -(B1$="")GOTO702:IFB1$="D"
THENB1=2:GOTO760                        OK
•704 B1=1:PRINT"[CLEAR]"                BO
•708 PRINT:PRINT:PRINT"LOAD CONTINUATION"
:PRINT                                  FE
•710 OPEN1,1,0,"CONTINUATION"           NM
•720 INPUT#B1,SP,OO,GS,X,Y,Z,R(3),R(4),R(
6),R(7)                                 NP
•730 FORI=2TO7:INPUT#B1,S(I):NEXT       GN
```

```
•735 FORI=2TO7:INPUT#B1,D$(I):NEXT                AA
•740 FORC=0TO5:FORB=0TO5:FORA=0TO5:INPUT#
 B1,R$(A,B,C):NEXTA,B,C                           CM
•750 CLOSEB1:IFB1=1THEN:GOTO90                    AP
•752 OPEN15,8,15:INPUT#15,E1$,E2$,E3$,E4$
 :CLOSE15:E1=VAL(E1$):IFE1<20THENGOTO90           LM
•754 PRINT"[CLEAR][DOWN][DOWN]DISK ERROR"
 :PRINTE1$,E2$,E3$,E4$:END                        DK
•760 PRINT:PRINT"LOADING PLEASE WAIT[4"."
 ]"                                               BC
•762 OPEN2,8,2,"0:CONTINUATION,S,R":GOTO7
 20                                               AA
•800 POKEV+21,0:PRINT"[CLEAR][c 8]":PRINT
 :PRINT" DO YOU WISH TO                           IN
•801 PRINT:PRINT"[c M][RVSON]C[RVSOFF]ONT
 INUE THIS GAME OR[c M][RVSON]N[RVSOFF]OT
 ?":PRINT:PRINT                                   BK
•805 GETA$:IFA$=""THEN805                         KC
•806 IFA$<>"C"THENEND                             BO
•810 PRINT"SAVE TO [RVSON]T[RVSOFF]APE OR
  [RVSON]D[RVSOFF]ISK?":PRINT:PRINT               CH
•812 GETB1$:ON -(B1$="")GOTO812:IFB1$="D"
 THENB1=2:GOTO870                                 FE
•814 B1=1:PRINT"[CLEAR]"                          BO
•820 PRINT"SAVE CONTINUATION":PRINT:PRINT HE
•825 OPEN1,1,1,"CONTINUATION":C$=CHR$(13) PO
•830 PRINT#B1,SP;C$;OO;C$;GS;C$;X;C$;Y;C$
 ;Z                                               GI
•835 PRINT#B1,R(3);C$;R(4);C$;R(6);C$;R(7
 )                                                NH
•840 FORI=2TO7:PRINT#B1,S(I):NEXT                 LB
•845 FORI=2TO7:PRINT#B1,D$(I):NEXT                OE
•850 FORC=0TO5:FORB=0TO5:FORA=0TO5:PRINT#
 B1,R$(A,B,C):NEXTA,B,C                           CA
•860 PRINT#B1:CLOSEB1:IFB1=1THEN END              MG
•862 PRINT#15,"V":CLOSE15:END                     OE
•870 PRINT:PRINT"SAVING TO DISK[4"."]"            OG
•872 OPEN15,8,15,"S0:CONTINUATION":OPEN2,
 8,2,"0:CONTINUATION,S,W":C$=CHR$(13)             MJ
•874 GOTO830                                      CL
•1000 NY=0:NX=0:IFS(3)>0ORS(7)>0THENGOSUB
 1600                                             CN
•1005 P=PEEK(VC):IFP=2THEN1030                    EC
•1008 GETM$:IFM$<>""THEN1040                      IN
•1010 AY=AY+NY*ABS(P<>2):AX=AX+NX*ABS(P<>
 2):POKEV2,AX:POKEV3,AY                           LM
•1015 ND=-(AY<65)-2*(AX>150)-3*(AY>175)-4
 *(AX<35):IFND>0THENRETURN                        HN
•1020 GOTO1005                                    FF
•1030 AX=AX-NX:AY=AY-NY:POKEV2,AX:POKEV3,
 AY                                               BI
•1040 NY=SP*(M$="X")-SP*(M$="Z"):NX=SP*(M
 $=",")-SP*(M$=".")                               LP
•1045 IFM$="[F1]"THEN800                          LJ
•1050 GOTO1010                                    FB
•1100 R$=R$(X,Y,Z):FORI=1TO4:PD(I)=VAL(MI
 D$(R$,I+1,1))                                    BG
•1115 IF(S(3)>0ORS(7)>0)ANDI=KTHENPD(I)=1 KC
•1116 PRINTPD$(I);:ONPD(I)+1GOTO1120,1130
 ,1140,1145                                       FP
•1120 PRINTWD$(I):GOTO1147                        HK
•1130 PRINTOD$(I):GOTO1147                        IC
•1140 PRINTGD$(I):GOTO1147                        GK
•1145 PRINTSD$(I)                                 PH
•1147 NEXT:PRINTLP$;L$(Z)                         GK
•1150 PRINTQP$SPC(X1);:Q$=" ":ONY1+1GOSUB
 1170,1180,1190                                   NG
•1151 X1=INT(X/2):Y1=INT(Y/2):X2=X-X1*2:Y
 2=Y-Y1*2                                         CE
•1152 QS=(X2=0ANDY2=0)+2*(X2=1ANDY2=0)+3*
 (X2=0ANDY2=1)+4*(X2=1ANDY2=1)                    OO
•1160 Q$=Q$(ABS(QS))                              MF
•1165 PRINTQP$SPC(X1);:ONY1+1GOSUB1170,11
 80,1190:RETURN                                   LP
•1170 PRINTQ$:RETURN                              OG
•1180 PRINT"[DOWN]"Q$:RETURN                      CL
•1190 PRINT"[DOWN][DOWN]"Q$:RETURN                EK
•1200 E$=LEFT$(R$(X,Y,Z),1):IFE$="0"THENR
 ETURN                                            GD
•1210 E=ASC(E$)-48:FORU=1TO10:POKESE+U,0:
 NEXT                                             IN
•1220 ONEGOTO2100,2200,2300,2400,2500,260
 0,2800                                           JO
•1225 ONE-16GOTO3100,3200,2900,3300,3200,
 3200                                             FA
•1230 E=E-23:GOTO2700                             AD
•1300 PRINT"[HOME][3"[DOWN]"[4"[RIGHT]"][
 GREEN]";:FORM=1TO14:PRINT"[14" "]"SPC(26
 );:NEXT:RETURN                                   PA
•1400 FORU=0TO10:POKESE+U,0:NEXT                  HJ
•1410 POKESE,80:POKESE+1,33:POKESE+5,72:P
 OKESE+22,104:POKESE+4,33                         AI
•1420 FORO=1TO200:NEXT:POKESE+4,32:RETURN HD
•1500 D$(L-76)=RIGHT$(STR$(X),1)+RIGHT$(S
 TR$(Y),1)+RIGHT$(STR$(Z),1):RETURN               PH
•1600 IFS(7)>0THENDN=3:GOSUB1680                  NP
•1610 IFS(3)>0THENDN=2:GOSUB1680                  CO
•1620 RETURN                                      IM
•1650 FORU=0TO10:POKESE+U,0:NEXT                  HJ
•1660 POKESE+1,60:POKESE+5,128:POKESE+4,1
 29                                               NB
•1670 FORO=0TO50:NEXT:POKESE+4,128:RETURN GP
•1680 FORI=1TO4:IFPD(I)=DNTHENPRINTPD$(I)
 OD$(I):GOSUB1650                                 LN
•1690 NEXT:RETURN                                 EJ
•1700 PD$(1)="[HOME][8"[RIGHT]"[3"[DOWN]"
 ]":PD$(2)="[HOME][17"[RIGHT]"[9"[DOWN]"]
 "                                                BK
•1710 PD$(3)="[HOME][17"[DOWN]"[9"[RIGHT]
 "]":PD$(4)="[HOME][8"[DOWN]"[3"[RIGHT]"]
 "                                                LG
•1715 OD$(1)="[4" "]":OD$(3)=OD$(1):OD$(2
 )=" [LEFT][DOWN] [LEFT][DOWN] [LEFT][DOW
 N] ":OD$(4)=OD$(2)                               NC
•1720 WD$(1)="[c 1][RVSON][c E][c R][s +]
 [c E][RVSOFF]":WD$(3)=WD$(1):WD$(2)="[c
```

```
1][RVSON][c Q][DOWN][LEFT][s +][DOWN][LE
FT][c W][DOWN][LEFT][c Q][RVSOFF]":WD$(4
)=WD$(2)                                    KG
•1725 GD$(1)="[c 6][RVSON][4"[c P]"][RVSO
FF]":GD$(2)="[RVSON][c 6][c G][DOWN][LEF
T][c G][DOWN][LEFT][c G][DOWN][LEFT][c G
][RVSOFF]":GD$(3)="[RVSON][c 6][4"[c Y]"
][RVSOFF]"                                  PA
•1730 GD$(4)="[RVSON][c 6][c N][DOWN][LEF
T][c N][DOWN][LEFT][c N][DOWN][LEFT][c N
][RVSOFF]"                                  GD
•1735 SD$(1)="[WHITE][RVSON][4"[c P]"][RV
SOFF]":SD$(2)="[RVSON][WHITE][c G][DOWN]
[LEFT][c G][DOWN][LEFT][c G][DOWN][LEFT]
[c G][RVSOFF]":SD$(3)="[RVSON][WHITE][4"
[c Y]"][RVSOFF]"                            BL
•1740 SD$(4)="[RVSON][WHITE][c N][DOWN][L
EFT][c N][DOWN][LEFT][c N][DOWN][LEFT][c
 N][RVSOFF]":RETURN                         IN
•1900 FORU=1TO23:POKESE+U,0:NEXT:POKESE+1
,130:POKESE+5,9:POKESE+15,30                DB
•1910 POKESE+4,21:FORT=1TO500:NEXT:POKESE
+4,20:POKESE+24,15:RETURN                   EM
•2000 E$(1)="[HOME][WHITE][13"[DOWN]"[6"[
RIGHT]"[3"[c @]"][DOWN][3"[LEFT]"][c M][
s M][c M][s M][DOWN][LEFT][LEFT][s O][s
P]"                                         OO
•2004 EL$="[DOWN][12"[LEFT]"]"              CO
•2020 E$(2)="[HOME][3"[DOWN]"[5"[RIGHT]"]
[YELLOW][s U][s I][s U][s I][DOWN][4"[LE
FT]"[4"[s -]"]MACDUCHESS"+EL$               OH
•2021 E$(2)=E$(2)+"[LEFT][s G][s H][s G][
s H][c 3][c L][RVSON] OVER 6[RVSOFF][c K
]"+EL$+"[YELLOW][LEFT][s T][s Y][s T][s
Y][c 3][c L][RVSON]BILLION[RVSOFF][c K]"
+EL$                                        EJ
•2022 E$(2)=E$(2)+"[LEFT][YELLOW][c G][c
M][c G][c M][c 3][c L][RVSON] SERVED[RVS
OFF][c K]"                                  PJ
•2030 E$(5)="[HOME][3"[DOWN]"[7"[RIGHT]"]
[c 5][8"[c P]"][DOWN][9"[LEFT]"][c M][RV
SON]  MAD[3" "][RVSOFF][c G][DOWN][10"[L
EFT]"]"                                     NK
•2031 E$(5)=E$(5)+"[c M][RVSON]HATTER'S[R
VSOFF][c G][DOWN][10"[LEFT]"][c M][RVSON
]TEAHOUSE[RVSOFF][c G][DOWN][10"[LEFT]"]
"                                           FH
•2032 E$(5)=E$(5)+"[LEFT][RVSON][12"[c I]
"][RVSOFF]"                                 LO
•2050 E$(6)="[HOME][BLUE][3"[DOWN]"[5"[RI
GHT]"][c D][11"[c I]"][c F][DOWN][13"[LE
FT]"][RVSON][c K]CARPENTER'S[RVSOFF][c K
][DOWN][13"[LEFT]"]"                        HK
•2051 E$(6)=E$(6)+"[RVSON][c K][s I] SEAF
OOD [s U][RVSOFF][c K][DOWN][13"[LEFT]"]
[RVSON][c K][s J]  DINER  [s K][RVSOFF][
c K]"                                       LK
•2099 RETURN                                IM
•2100 PRINT"[HOME][c 3][4"[DOWN]"[5"[RIGH
T]"][RVSON][c A][6"[s *]"][c S][DOWN][LE
FT][s -][DOWN][LEFT][s -][DOWN][LEFT][s
-][DOWN][LEFT][s -][DOWN][LEFT][s -][DOW
N][LEFT][s -][DOWN][LEFT][c X][LEFT][LEF
T][s *][6"[LEFT]"][s *][LEFT][LEFT][c Z]
";                                          HI
•2105 PRINT"[UP][LEFT][s -][UP][LEFT][s -
][UP][LEFT][s -][UP][LEFT][s -][UP][LEFT
][s -][UP][LEFT][s -]":GOSUB600:SK=-1       DF
•2110 IFS(2)=0THENPRINTSP$"SORRY ALICE, B
UT YOU NEED A KEY"                          GA
•2111 IFS(2)=0THENFORA=1TO1200:NEXT:GOSUB
1300:RETURN                                 JK
•2115 PRINTSP$"DO YOU WANT TO USE THE ELE
VATOR?"                                     AJ
•2120 GETA$:IFA$=""THEN2120                 NP
•2121 IFA$<>CHR$(32)THENE=0:PRINTER$:GOSU
B1300:RETURN                                OP
•2125 PRINTER$SP$"WHAT LEVEL?"              KB
•2140 POKEV+2,PX(5)-16:POKEV+3,PY(5)-16     CN
•2142 GETA$:IFA$=""THEN2142                 NP
•2145 IFASC(A$)>48ANDASC(A$)<55THENNL=VAL
(A$)-1:PRINTER$:GOTO2160                    MB
•2150 GOTO2142                              FK
•2160 FORL=ZTONLSTEP.1*(1+2*(Z>NL)):POKES
E+4,17                                      MN
•2180 POKESE+5,8:POKESE+6,8:POKESE+1,L*20
:POKESE,250-L*5:PRINTLP$;L$(INT(L)+1)       EL
•2190 FORO=1TO25:NEXT::POKESE+4,0:NEXT:ND
=0:Z=NL:SK=-1                               AC
•2195 AX=PX(5)-16:AY=PY(5)-16:GOSUB1300:S
3=S(3):S(3)=0:GOSUB1100:S(3)=S3:RETURN      IK
•2200 PRINT"[HOME]"TAB(3)"[c 8][3"[DOWN]"
][c I][DOWN][LEFT]";:FORO=1TO4:PRINT"[c
*][RVSON][c C][DOWN][LEFT][RVSOFF]";:NEX
T:Q=1:GOTO2315                              EK
•2250 FORL=25TO45+40*(Q=-1)STEP2+4*(Q=-1)
:POKESE+4,33                                ED
•2280 POKESE+5,8:POKESE+6,32:POKESE+1,L:P
OKESE,100-L                                 CD
•2290 FORD=1TO80:NEXT:POKESE+4,0:NEXT:ND=
0:K=5:RETURN                                FB
•2300 PRINT"[HOME]"TAB(12)"[c 8][12"[DOWN
]"][c I][DOWN][LEFT][RVSOFF]";:FORO=1TO4
:PRINT"[c *][RVSON][c C][DOWN][LEFT][RVS
OFF]";:NEXT:Q=-1                            BC
•2315 GOSUB600:PRINTSP$"DO YOU WISH TO US
E THE STAIRS?"                              LD
•2320 GETA$:IFA$=""THEN2320                 LJ
•2330 IFA$<>CHR$(32)THENPRINTER$:SK=-1:RE
TURN                                        BL
•2345 Z=Z+Q:PRINTER$:S(1)=0:POKEV+21,FNR(
R):GOSUB2250:POKEV2,PX(5):POKEV3,PY(5)      BF
•2347 K=5:AX=PX(K):AY=PY(K):ND=0:GOSUB130
0:E=0:GOSUB1100:S(1)=2:SK=-1                HF
•2350 POKEV+21,FNR(R):RETURN                HO
•2360 GOTO2320                              FA
```

```
•2400 READL:FORI=LTOL+62:READJ:POKEI,J:NE
XT:POKE2040,15:S(0)=1:POKEV,PX(1)+1          PM
•2410 POKEV+1,PY(1)+15:POKEV+39,12:POKEV+
28,3:POKEV+21,FNR(R):GOSUB600                NH
•2420 PRINTSP$"ALICE!! YOU FOUND THE WHIT
E RABBIT!":GOSUB2460                         HH
•2450 GOSUB1900:PRINT"[4"[UP]"]":END         GF
•2460 FORL=1TO15:POKESE+4,33                 HI
•2480 POKESE+5,8:POKESE+6,32:POKESE+1,30:
POKESE,100                                   PN
•2490 FORD=1TO40:NEXT:POKESE+4,32            IC
•2495 POKESE+5,64:POKESE+6,8:POKESE+1,20:
FORC=1TO40:POKESE,C:NEXT:NEXT:RETURN         EA
•2500 PRINT"[HOME][4"[DOWN]"[5"[RIGHT]"][
c 7][RVSON][c D]"SPC(10)"[c F][LEFT][11"
[DOWN]"][c V][12"[LEFT]"][c C]":IFS(5)>0
THENRETURN                                   CA
•2503 GOSUB600:PRINT"[HOME][5"[DOWN]"[6"[
RIGHT]"][s M][DOWN][s M][DOWN][s M][DOWN
][s M][DOWN][s M][DOWN][s M][DOWN][s M][
DOWN][s M][DOWN][s M][DOWN][s M][10"[LEF
T]"]";                                       PD
•2505 PRINT"[s N][UP][s N][UP][s N][UP][s
 N][UP][s N][UP][s N][UP][s N][UP][s N][
UP][s N][UP][s N]":POKEV+2,PX(5):POKEV+3
,PY(5)                                       PN
•2510 ND=0:GOSUB2550:GOSUB2580:Z=Z-1:ND=0
:SK=-1:IFGS=0THEN2535                        AL
•2530 P=RND(0):IFP<.7THENPOKEG2+GS,32:GS=
GS-ABS(GS<>0):GOSUB2640                      LD
•2532 IFP<.7THENPRINTSP$"YOU LOST A GOLD
COIN!":FORI=1TO700:NEXT                      ON
•2535 AX=PX(5):AY=PY(5):PRINTER$             OG
•2540 GOSUB1300:S3=S(3):S(3)=0:S7=S(7):S(
7)=0                                         MA
•2545 GOSUB1100:S(3)=S3:S(7)=S7:GOSUB1200
:RETURN                                      KJ
•2550 FORU=ZTOZ-1STEP-.2:POKESE+4,33         PP
•2560 POKESE+5,8:POKESE+6,8:POKESE+1,U*20
:POKESE,250-U*5                              HH
•2565 PRINTLP$;L$(INT(U+1))                  IA
•2570 FORD=1TO25:NEXT:POKESE+4,0:NEXT:RET
URN                                          LA
•2580 FORU=0TO23:POKESE+U,0:NEXT             CF
•2585 POKESE,80:POKESE+1,33:POKESE+5,72:P
OKESE+22,104:POKESE+4,129                    GD
•2590 POKESE+24,79:FORO=1TO350:NEXT:POKES
E+4,128:POKESE+24,15:GOSUB1300:RETURN        HB
•2600 PRINT"[HOME][6"[DOWN]"[7"[RIGHT]"][
PURPLE][RVSON][s +][6"[RIGHT]"][s +][LEF
T][7"[DOWN]"][s +][8"[LEFT]"][s +][RVSOF
F]"                                          NL
•2603 IFS(4)>0THENRETURN                     JN
•2604 GOSUB600:PRINTSP$"LOOKING GLASS ROO
M!":GOSUB2680:FORU=1TO200:NEXT               IC
•2605 POKEV+2,PX(5):POKEV+3,PY(5):PRINT"[
HOME][PURPLE][7"[DOWN]"]"TAB(8);             LC
•2607 X=INT(5*RND(0)):Y=INT(5*RND(0)):Z=I
NT(5*RND(0))                                 PB
•2608 IFX=1ANDY=5AND(Y=3ORY=4)THEN2607       LL
•2610 FORI=1TO4:PRINT"[s W]";:GOSUB2640:N
EXT:PRINT"[DOWN]";:FORI=1TO4:PRINT"[s W]
[DOWN][LEFT]";                               GK
•2615 GOSUB2640:NEXT:PRINT"[LEFT]";:FORI=
1TO4:PRINT"[s W][LEFT][LEFT]";:GOSUB2640
:NEXT:PRINT"[UP]";                           PA
•2620 FORI=1TO4:PRINT"[s W][UP][LEFT]";:G
OSUB2640:NEXT:S(1)=0:POKEV+21,FNR(R):ND=
0:K=5                                        LC
•2625 AX=PX(5):AY=PY(5)                      GA
•2630 GOSUB2680:FORT=1TO200:NEXT:S(1)=2:P
OKEV+21,FNR(R)                               GL
•2632 FORT=1TO200:NEXT:GOSUB1300:PRINTER$    NN
•2635 S3=S(3):S(3)=0:S7=S(7):S(7)=0:GOSUB
1100:S(3)=S3:S(7)=S7                         DO
•2637 GOSUB1200:SK=-1:RETURN                 JI
•2640 POKESE+5,128:POKESE+4,33               NE
•2650 POKESE+1,100:POKESE,45:FORT=1TO99:N
EXT:POKESE+4,32:FORT=1TO90:NEXT:RETURN       DB
•2680 POKESE,210:POKESE+1,140:POKESE+5,15
:POKESE+4,129                                CL
•2690 FORO=0TO150:NEXT:POKESE+4,32:RETURN    KA
•2700 F=E-5:POKEV+SP(E),PX(5):POKEV+SP(E)
+1,PY(5):S(F)=2[UPARROW](F):R(F)=0           AC
•2705 POKEV+21,FNR(R):POKEV+16,FNX(R):R$(
X,Y,Z)="0"+RIGHT$(R$(X,Y,Z),4):GOSUB600      CN
•2710 PRINTSP$"YOU FOUND THE "K$(E)"!":FO
RI=1TO600:NEXT:R(F)=S(F)                     OO
•2715 POKEV+16,FNX(R):POKEV+SP(E),PX(E)      CO
•2720 POKEV+SP(E)+1,PY(E):GOSUB1900:SK=-1
:RETURN                                      PN
•2800 S(0)=1:POKEV+1,PY(4)-10:POKEV+30,0:
POKEV,0                                      IG
•2805 POKEV+28,3:POKEV+39,7:POKE2040,11:P
OKEV+21,FNR(R):GOSUB600                      NK
•2808 IF(X=0ANDY=0)OR(X=1ANDY=5AND(Z=3ORY
=4))THEN2607                                 GM
•2810 O=00:SL=INT(6*RND(0))+7:IFSL=11THEN
SL=12                                        BF
•2811 FORI=5TO155STEP5:POKEV,I               JP
•2812 POKESE+1,160:POKESE+5,64:POKESE+4,1
29                                           AJ
•2814 FORJ=0TO10:NEXT:POKESE+4,128           NP
•2815 IF(PEEK(V+30)AND3=3)THENGOSUB2870      GD
•2820 NEXT:POKEV+21,FNR(R)                   HK
•2825 IFSV<>0THENR$(A(O),B(O),C(O))=CHR$(
SV+76)+RIGHT$(R$(A(O),B(O),C(O)),4)          OH
•2830 IFSV<>0THENPRINTSP$"THE GRYPHON STO
LE YOUR [c 8]"K$(SV+5)"[CYAN]!"              EA
•2832 A$=RIGHT$(STR$(A(O)),1):B$=RIGHT$(S
TR$(B(O)),1):C$=RIGHT$(STR$(C(O)),1)         FF
•2835 S(0)=0:POKEV+21,FNR(R):D$(SV)=A$+B$
+C$:SV=0:SK=-1:RETURN                        GM
•2870 IFS(SL-5)=0ORS(6)>0THENRETURN          FN
•2875 GOSUB2640:SV=SL-5:S(SL-5)=0:R(SL-5)
=0:POKEV+21,FNR(R):OO=OO+1+15*(OO=15)        DH
```

```
•2880 RETURN                                        IM
•2900 POKEV+28,2:POKEV+39,8:POKE2040,15             IK
•2910 S(0)=1:POKEV+21,FNR(R):POKEV+1,PY(5
 ):POKEV,PX(5):GOSUB600:SK=-1                       AM
•2920 FORI=1TO3:F(I)=VAL(MID$(D$(Z+2),I,1
 )):NEXT                                            JN
•2930 DR=(F(1)<3ANDF(2)<3)+(F(1)>2ANDF(2)
 <3)*2+(F(1)<3ANDF(2)>2)*3                          LM
•2935 DR=DR+4*(F(1)>2ANDF(2)>2)                     CK
•2950 IFS(Z+2)>0THENPRINTSP$"'I SEE YOU H
 AVE THE [c 8]"K$(Z+7)"[CYAN]!'":RETURN             KD
•2952 PRINTSP$"'YOU'LL FIND THE [c 8]"K$(
 Z+7)                                               EO
•2955 PRINTSP$"[DOWN]IN THE [YELLOW]"DR$(
 ABS(DR))"[CYAN] SECTION ON LEVEL[YELLOW]
 "STR$(F(3)+1)"[CYAN]'":RETURN                      BK
•3100 PRINTE$(E-16):E=0:RETURN                      OC
•3200 PRINTE$(E-16):SK=-1:GOSUB600:IFGS>0
 THENPRINTSP$;ST$(E-16)                             BO
•3202 E=0:IFGS=0THENPRINTSP$"SORRY, LITTL
 E GIRL, BUT YOU NEED GOLD":RETURN                  EE
•3205 POKEG2+GS,32:GS=GS-1:GOSUB2640:RETU
 RN                                                 KK
•3300 POKEG1,81:POKEG1+LC,7:GOSUB600:PRIN
 TSP$"YOU FOUND A GOLD COIN!":SK=-1                 OP
•3310 GS=GS+ABS(GS>-1):POKEG1,32:POKEG2+G
 S,81:POKEG2+GS+LC,7:GOSUB1900                      OC
•3320 R$(X,Y,Z)="0"+RIGHT$(R$(X,Y,Z),4)             MN
•3330 IFGS<10THEN:RETURN                            LI
•3335 GS=0:FORJ=2TO7:IFS(J)=0THEN3350               BG
•3340 NEXTJ:FORI=10TO1STEP-1:POKEG2+I,32:
 GOSUB2640:NEXT                                     DE
•3342 PRINTSP$"ALICE! THE WHITE RABBIT IS
  IN"                                               AK
•3345 PRINTSP$"[DOWN]THE [YELLOW]"DR$(3)"
 [CYAN] SECTION ON LEVEL[YELLOW]"A(11):RE
 TURN                                               LO
•3350 S(J)=2[UPARROW]J:POKEV+SP(J+5),PX(J
 +5):POKEV+SP(J+5)+1,PY(J+5):R(J)=S(J)              IB
•3355 POKEV+16,FNX(R):FORI=10TO1STEP-1:PO
 KEG2+I,32:GOSUB2640:NEXT                           AD
•3360 A=VAL(LEFT$(D$(J),1)):B=VAL(MID$(D$
 (J),2,1)):C=VAL(RIGHT$(D$(J),1))                   OA
•3365 R$(A,B,C)="0"+RIGHT$(R$(A,B,C),4):P
 OKEV+21,FNR(R):GOSUB1900                           AJ
•3370 PRINTER$SP$"YOU JUST BOUGHT THE [c
 8]"K$(J+5)                                         KM
•3380 R$(X,Y,Z)="0"+RIGHT$(R$(X,Y,Z),4):R
 ETURN                                              KE
•9000 DATA" [c S][DOWN][LEFT][LEFT] [s -]
 [DOWN][LEFT][LEFT] [c E]","[s U][s I][DO
 WN][LEFT][LEFT][s U][s K][DOWN][LEFT][LE
 FT][c Z][c X]","[s U][s I][DOWN][LEFT][L
 EFT] [c W][DOWN][LEFT][LEFT][s J][s K]"            EB
•9005 DATA"[s U][c S][DOWN][LEFT][LEFT][c
  Z][s +][DOWN][LEFT][LEFT] [c E]","[c A]
 [c S][DOWN][LEFT][LEFT][c Z][s I][DOWN][
 LEFT][LEFT][s J][s K]","[s U][s I][DOWN]
 [LEFT][LEFT][c Q][s I][DOWN][LEFT][LEFT]
 [s J][s K]"                                        FC
•9100 DATANW,NE,SW,SE                               IG
•9200 DATA"[c V]","[c C]","[c F]","[c D]"           OL
•9550 DATA92,68,152,120,92,172,38,120,92,
 120,92,160,237,120,12,120,45,120                   KG
•9551 DATA237,152,12,152,45,152                     BP
•9560 DATA ELEVATOR KEY,GREEN KEY, LOOKIN
 GGLASS KEY                                         KN
•9561 DATAKEY FOR THE PITS,GRYPHON KEY,WH
 ITE KEY                                            OK
•9570 DATA0,4,0,3,0,3,1,1,4,1,5,1,5,4,2,1
 ,5,5,5,5,4,1,1,0,5,0,2,5,5,3                       PN
•9571 DATA4,3,0,5,0,5,0,5,4,2,0,2,5,4,1             IP
•9580 DATA2,HAVE A HAMBURGER AND FRIES!,5
 ,YOU MUST BE THIRSTY!                              ID
•9581 DATA6,EAT AN OYSTER FOR DINNER!               DE
•9610 DATA10110,00121,D0111,00011,D0030,0
 0000                                               DP
•9611 DATA01120,D2001,01110,71101,63201,R
 0002                                               GA
•9612 DATA02130,E0121,01211,00112,00111,0
 0011                                               HD
•9613 DATA03120,62211,C1102,01021,D1000,0
 1010                                               LG
•9614 DATAD2130,21211,D0202,02112,00111,0
 1021                                               IM
•9615 DATA03000,01100,70101,D1001,A1000,0
 2000                                               FK
•9620 DATA10120,00011,50030,50110,20111,0
 0011                                               KE
•9621 DATA02200,01012,63100,D1111,01021,0
 1010                                               PE
•9622 DATAB0100,01111,00101,C1201,02112,0
 1021                                               IG
•9623 DATAD0000,D1110,70121,00211,01112,N
 2011                                               GH
•9624 DATA00110,31201,02002,01110,01011,D
 1020                                               OM
•9625 DATA01100,D0301,00103,D1101,51101,D
 2001                                               LE
•9630 DATA10110,00201,D0102,00011,30010,D
 0010                                               JC
•9631 DATA01110,D0101,C0111,01101,01111,0
 1011                                               ON
•9632 DATA51220,D0012,01110,70121,01011,6
 1000                                               FH
•9633 DATA22110,01101,01211,D2102,01131,F
 0011                                               OL
•9634 DATAD1120,50111,01001,00000,03210,D
 1002                                               GH
•9635 DATA02000,01200,S0002,D0100,51301,B
 0003                                               AK
•9640 DATA10110,00101,00111,D0101,60311,D
 0003                                               HE
•9641 DATA01220,C0012,01010,00120,01001,5
 0010                                               OI
•9642 DATAD2010,01110,D1111,02221,00312,D
```

```
1013                                          KN
•9643 DATA31120,01011,51220,D2102,71101,0
1011                                          JE
•9644 DATA62110,01101,02101,20121,F0121,0
1011                                          AA
•9645 DATA01000,40000,E0300,52103,02201,D
1002                                          CO
•9650 DATA10110,00101,20011,D0000,50130,F
0001                                          KF
•9651 DATA01010,D0100,01111,70101,03201,Q
0022                                          EF
•9652 DATA01110,50201,01122,D0121,E0101,0
2011                                          HC
•9653 DATA01120,D0101,02211,D2222,60102,D
1011                                          MK
•9654 DATAC2110,70131,01011,32110,00111,0
1011                                          KF
•9655 DATAD1000,53000,01100,01201,D1002,0
1000                                          BJ
•9660 DATA10120,00011,30110,B0111,00111,D
0021                                          KE
•9661 DATA02110,01121,01111,D1001,01020,P
2010                                          HL
•9662 DATA01210,62132,01121,00211,02212,0
1012                                          PG
•9663 DATA01010,D3000,C2020,01110,01031,D
1010                                          EK
•9664 DATA01110,00111,02101,71121,03111,0
1001                                          GO
•9665 DATA51200,D1002,00000,D2100,51301,D
0003                                          GD
•9700 DATA832,0,20,0,0,125,0,0,125,0,0,60
,0,0,170,0,2,170,128,10,40,160                AP
•9701 DATA40,40,40,240,40,15,0,150,0,2,15
0,128,2,150,128,10,150,160,10,170,160         HN
•9702 DATA10,170,160,0,60,0,0,60,0,0,60,0
,0,60,0,0,60,0,0,170,0                        IN
•9705 DATA896,0,0,0,0,0,0,0,24,0,0,60,0,0
,102,0,0,102,0,0,126,0                        OB
•9706 DATA0,60,0,0,24,0,0,24,0,0,24,0,0,2
4,0,0,24,0,0,24,0                             AF
•9707 DATA0,88,0,0,120,0,0,24,0,0,120,0,0
,88,0,0,0,0,0,0,0                             NL
•9715 DATA704,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,248,0,168              NN
•9716 DATA170,128,168,255,160,175,42,168,
47,63,250,40,0,170,160,170,170,144            BH
•9717 DATA42,229,64,7,212,0,3,192,0,15,0,
0,3,192,0,0,252,0                             HH
•9720 DATA960,0,0,0,0,0,0,0,0,0,33,0,0,51
,7,240,127,159,252,255,223,254,204,223        EE
•9721 DATA255,255,223,31,222,223,127,225,
223,127,127,223,255,65,63,255,125,255         KA
•9722 DATA207,125,231,63,62,240,255,0,255
,254,0,63,248,0,0,0,0,0,0,0,0,0               FG
•9800 DATA960,0,68,0,0,68,0,0,68,0,0,84,0
,0,220,0,0,116,0                              JH
•9802 DATA0,84,0,0,16,0,0,152,0,2,170,0,1
0,170,128,8,136,128                           NM
•9804 DATA8,168,128,4,136,64,0,168,0,1,85
,0,1,69,0,1,69,0                              MP
•9806 DATA0,68,0,0,68,0,0,68,0                GJ
```

Music Tutor

FROM PAGE 37

```
•5 PRINT"[CLEAR][DOWN] PLEASE WAIT":V=532
48                                            LN
•10 DIMA(21),B(21):POKEV+32,15:POKEV+33,1
1                                             LO
•20 KB$="GABCDEFGABCDEFGABCDEF"               CK
•30 FORX=12288TO12928STEP64:FORY=0TO62:RE
ADA:POKEX+Y,A:NEXTY,X                         HA
•40 FORX=1TO21:READA,B:A(X)=A:B(X)=B:NEXT     PK
•50 POKE2040,192:POKE2041,193:POKE2045,19
7:POKE2047,199                                OD
•60 POKEV+37,2:POKEV+38,1:POKEV+39,0:POKE
V+40,0:POKEV+44,1:POKEV+46,1                  EJ
•70 POKEV,50:POKEV+1,138:POKEV+2,58:POKEV
+3,190                                        FF
•80 POKEV+10,177:POKEV+11,98:POKEV+14,172
:POKEV+15,74                                  AD
•90 POKEV,50:POKEV+1,138:POKEV+2,58:POKEV
+3,190:POKEV+23,128                           GJ
•95 GOTO900                                   CN
•100 FORX=1TO2:PRINT"[c 7][UP][c H][RVSON
][c G][RVSOFF] [RVSON] [s -] [RVSOFF] [R
VSON] [RVSOFF] [RVSON] [s -] [RVSOFF] [R
VSON] [RVSOFF] [RVSON] [RVSOFF] [RVSON]
[s -][c 6] [RVSOFF] [RVSON] [RVSOFF] [RV
SON] [s -] [RVSOFF] [RVSON] [RVSOFF] [RV
SON] [RVSOFF] [
RVSON] ";                                     HD
•105 PRINT"[RVSON][s -] [RVSOFF] [RVSON]
[RVSOFF] [RVSON] [s -] ":NEXT                 HC
•110 FORX=1TO2:PRINT"[UP][RVSON][c 7][s Y
] [s -][SS][s -][SS][s -][SS][s -][SS][s
 -][SS][s -][SS][s -][SS][s -][SS][s -][
c 6][SS][s -][SS][s -][SS][s -] [s -][SS
][s -][SS][s -][SS][s -][SS][s -][SS][s
-][SS][s -][SS]":NEXT                         BN
•115 PRINT:PRINT:PRINT"[WHITE]"               IE
•120 PRINT"[c 6][3" "][c A]";:FORX=1TO30:
PRINT"[s *]";:NEXT:PRINT"[c R]";:PRINT"[
c S]"                                         CD
•125 FORY=1TO4:PRINT"[3" "][c Q]";:FORX=1
TO30:PRINT"[s *]";:NEXT:PRINT"[s +]";:PR
INT"[c W]":NEXT                               CO
•130 PRINT"[WHITE][3" "][s -]";:FORX=1TO1
5:PRINT" ";:NEXT                              HD
•135 PRINT"[3"[s C]"]";:FORX=1TO12:PRINT"
 ";:NEXT:PRINT"[s -]";:PRINT"[s -]":PRIN
T"[c 7]";                                     BL
•140 FORY=1TO4:PRINT"[3" "][c Q]";:FORX=1
TO30:PRINT"[s *]";:NEXT:PRINT"[s +]";:PR
INT"[c W]":NEXT                               CO
```

```
·145 PRINT"[3" "][c Z]";:FORX=1TO30:PRINT
 "[s *]";:NEXT:PRINT"[c E]";:PRINT"[c X]" KF
·150 IFG=1THEN1000                        CN
·160 POKE2042,194:POKE2043,194:POKE2044,1
 96:POKE2046,198                          FI
·170 POKEV+4,84:POKEV+5,143:POKEV+6,84:PO
 KEV+7,191:POKEV+12,165:POKEV+13,108      NN
·175 POKEV+28,64                          KG
·180 POKEV+41,0:POKEV+42,0:POKEV+43,7:POK
 EV+45,7                                  FK
·190 GOSUB4200:POKEV+29,79:POKEV+23,207:P
 OKEV+21,151                              CB
·200 A=INT(RND(1)*L1)+L2:IFA=1THENNI=1.5:
 GOTO210                                  HF
·205 NI=A                                 JB
·210 IFG>=3THENPRINT"[HOME][BLACK]"TAB(NI
 *2-3)"[WHITE][7"[DOWN]"][UPARROW]"       BN
·220 IFG<=3THENPOKEV+9,222-4*A            NJ
·230 POKEV+8,169:POKEV+21,159             JK
·240 A$=MID$(KB$,A,1)                     LJ
·250 GETN$:IFG=3ANDN$=""GOTO250           EH
·260 IFN$="[F1]"ORN$="[F3]"ORN$="[F5]"THE
 NGOSUB4200                               MA
·265 IFN$="[F7]"THENPOKEV+21,0:GOTO900    AL
·270 IFN$="H"THENGOSUB4000:GOTO250        LL
·275 ONG-1GOTO500,280,700                 FK
·280 IFASC(N$)<65ORASC(N$)>72THEN250      PH
·290 IFN$=A$THENGOSUB4500:GOTO400         HI
·300 GOSUB4600:GOTO250                    KH
·400 PRINT"[HOME]"TAB(NI*2-3)"[7"[DOWN]"]
 "                                        CP
·410 POKEV+21,143:GOTO200                 AD
·500 IFNP=50THENNP=19:PRINT"[HOME][BLACK]
 "TAB(NP)"[WHITE][7"[DOWN]"][UPARROW]":PN
 =NP                                      FP
·510 GOSUB3000:IFFR<>16THEN600            JL
·520 IFJV<>4ANDJV<>8THEN250               OF
·530 IFJV=8THENNP=NP+2:IFNP=2THENNP=1     PA
·540 IFNP=41THENNP=39:GOTO250             CB
·550 IFJV=4THENNP=NP-2:IFNP=-1THENNP=0    BN
·560 IFNP=-2THENNP=0:GOTO250              DI
·570 PRINT"[HOME]"TAB(PN)"[7"[DOWN]"] "   DI
·580 PRINT"[HOME][BLACK]"TAB(NP)"[WHITE][
 7"[DOWN]"][UPARROW]":PN=NP               EJ
·590 GOTO250                              CD
·600 N=INT(NP/2+2):IFNP=0THENN=1          JO
·620 IFN=ATHENGOSUB4500:GOTO410           AM
·630 GOSUB4600:GOTO250                    KH
·700 IFNP=50THENPOKEV+9,178:NP=11         OJ
·710 GOSUB3000:IFFR<>16THEN770            JN
·720 IFJV<>1ANDJV<>2THEN250               OE
·730 IFJV=1THENNP=NP+1:IFNP=22THENNP=21:G
 OTO250                                   AG
·740 IFJV=2THENNP=NP-1:IFNP=0THENNP=1:GOT
 O250                                     PJ
·750 POKEV+9,222-4*NP:GOTO710             EA
·770 IFNP=ATHENGOSUB4500:GOTO400          KB
·780 GOSUB4600:GOTO250                    KH
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
·900 PRINT"[CLEAR][DOWN][DOWN][CYAN][13"
 "]MUSIC TUTOR"                           GP
·910 PRINT"[3"[DOWN]"[6" "]1  NOTE PRIMER
 "                                        AP
·915 PRINT"[DOWN][5" "]2  KEYBOARD SEARCH
 "                                        MD
·920 PRINT"[DOWN][5" "]3  NAME THE NOTE"  LL
·925 PRINT"[DOWN][5" "]4  PIN THE NOTE ON
 THE STAFF"                               NF
·930 NP=50:L1=0                           EO
·935 PRINT"[4"[DOWN]"[14" "]PRESS CHOICE" CK
·940 GETA$:IFA$=""THEN940                 IA
·945 G=VAL(A$)                            IH
·950 IFG<1ORG>4THEN940                    OF
·955 ONGGOTO960,965,970,975               HB
·960 PRINT"[CLEAR][DOWN][14" "]NOTE PRIME
 R":PRINT:PRINT:POKEV+21,128:GOTO100      BL
·965 PRINT"[CLEAR][DOWN][12" "]KEYBOARD S
 EARCH":PRINT:PRINT:POKEV+21,128:GOTO100  AO
·970 PRINT"[CLEAR][DOWN][13" "]NAME THE N
 OTE":PRINT:PRINT:POKEV+21,128:GOTO100    DK
·975 PRINT"[CLEAR][DOWN][7" "]PIN THE NOT
 E ON THE STAFF":PRINT:PRINT:POKEV+21,128
 :GOTO100                                 LK
·1000 REM NOTE PRIMER                     JD
·1010 POKE2042,195:POKE2043,200:POKE2044,
 196:POKE2046,202                         GP
·1020 POKEV+23,131:POKEV+29,3:POKEV+28,0  NL
·1030 POKEV+4,170:POKEV+5,200:POKEV+6,186
 :POKEV+7,180                             GP
·1040 POKEV+8,170:POKEV+9,219:POKEV+12,22
 7:POKEV+13,139                           OC
·1050 POKEV+41,0:POKEV+42,0:POKEV+43,7:PO
 KEV+45,0                                 GB
·1055 GOSUB4250                           GG
·1060 PRINT"[YELLOW][3" "]MUSIC IS WRITTE
 N ON A SET OF LINES"                     IH
·1065 PRINT"[DOWN][6" "]AND SPACES KNOWN
 AS A [RVSON]STAFF[RVSOFF]."              NK
·1070 GOSUB4300:PRINT"[7" "]THE BASIC STA
 FFS ARE THE"                             NO
·1075 PRINT"[DOWN][4" "][RVSON][c 6]TREBL
 E[RVSOFF] [RVSON]CLEF[RVSOFF],[YELLOW]"; EP
·1080 FORX=1TO3:POKEV+21,129:FORY=1TO500:
 NEXT:POKEV+21,128:FORY=1TO100:NEXTY,X    AC
·1085 POKEV+21,129:FORX=1TO200:NEXT:PRINT
 " AND THE [RVSON][c 7]BASS[RVSOFF] [RVSO
 N]CLEF[RVSOFF].                          IF
·1090 FORX=1TO3:POKEV+21,131:FORY=1TO500:
 NEXT:POKEV+21,129:FORY=1TO100:NEXTY,X    NK
·1095 POKEV+21,131:GOSUB4300              FM
·1100 PRINTSPC(7)"TOGETHER THEY ARE KNOWN
 AS"                                      PF
```

```
•1103 PRINTTAB(11)"[DOWN]THE [RVSON][c 6]
GRAND[RVSOFF] [RVSON][c 7]STAFF[RVSOFF].
"                                        NM
•1105 FORX=1TO2:POKEV+21,128:FORY=1TO100:
NEXT:POKEV+21,131:FORY=1TO500:NEXTY,X   OM
•1110 GOSUB4300:PRINT" EACH LINE AND SPAC
E ON THE GRAND STAFF"                    IC
•1115 PRINT"[DOWN][5" "]REPRESENTS A KEY
ON THE PIANO"                            GC
•1120 POKE214,5:PRINT:PRINT"[WHITE][UPARR
OW]";:POKEV+21,147                       IO
•1121 POKE54296,15:POKE54275,0:POKE54274,
255:POKE54277,13:POKE54278,0             JK
•1122 POKE54273,A(1):POKE54272,B(1):POKE5
4276,65                                  HO
•1125 FORX=1TO1100:NEXT:PRINT"[WHITE][LEF
T] [UPARROW]";:POKEV+9,214:POKE54273,A(2
):POKE54272,B(2)                         IL
•1126 POKE54276,64:POKE54276,65          FI
•1130 FORY=1TO19:FORX=1TO1100:NEXT:PRINT"
[WHITE][LEFT]  [UPARROW]";:POKEV+9,PEEK(
V+9)-4                                   DN
•1131 POKE54273,A(Y+2):POKE54272,B(Y+2):P
OKE54276,64:POKE54276,65:NEXT            NH
•1132 FORX=1TO1000:NEXT:POKE54276,64     IO
•1135 GOSUB4300:POKEV+21,131             EK
•1140 PRINT"[DOWN][3" "]THE MUSIC ALPHABE
T IS FROM [RVSON]A[RVSOFF] TO [RVSON]G[R
VSOFF].":GOSUB4300                       DF
•1145 PRINT"  AFTER G IT REPEATS, STARTIN
G WITH A."                               CI
•1150 PRINTTAB(6)"[DOWN](A B C D E F G A
B C . . .)                               HI
•1155 GOSUB4300                          FM
•1160 PRINT"[YELLOW] EVERY LINE AND SPACE
 HAS A LETTER NAME"                      PK
•1170 PRINTTAB(8)"[DOWN]FROM THE MUSIC AL
PHABET"                                  BL
•1175 POKE214,20:PRINT:PRINTTAB(15)"[WHIT
E]G[UP]B[UP]D[UP]F[UP]A[UP]C[UP]E[UP]G[U
P]B[UP]D[UP]F"                           MM
•1180 POKEV+8,210:POKEV+9,160            FJ
•1185 POKE2044,201:POKEV+43,0:POKEV+44,11
:POKEV+45,0:POKEV+21,223                 CF
•1190 GOSUB4300:POKEV+21,255             EJ
•1195 PRINT"[UP][UP][RVSON][c 7]GA[s B]B[
s B]C[s B]D[s B]E[s B]F[s B]G[s B]A[s B]
B[s B] [c 6][s B]D[s B]E[s B]F[s B]G[s B
]A[s B]B[s B]C[s B]D[s B]E[s B]F"        OG
•1200 PRINT"[UP][YELLOW][6" "]THE NAMES O
F THE PIANO KEYS"                        NJ
•1205 PRINT"[DOWN][6" "]MATCH THE NAMES O
N THE STAFF"                             AC
•1210 GOSUB4300                          FM
•1215 PRINT"[4" "]TO HELP LEARN THE NAMES
 OF THE"                                 EI
•1220 PRINT"[DOWN][6" "]LINES AND SPACES,
 REMEMBER:":GOSUB4300                    MB
```

```
•1225 PRINTSPC(6)"TREBLE CLEF SPACES, THE
 WORD:"                                  NF
•1230 PRINTTAB(16)"[DOWN][RVSON][c 6]F[RI
GHT]A[RIGHT]C[RIGHT]E[RVSOFF]"           NF
•1240 GOSUB4300                          FM
•1245 PRINT"[4" "]TREBLE CLEF LINES, THE
SENTENCE:"                               BB
•1250 PRINT"[DOWN][5" "][RVSON][c 6]E[RVS
OFF]VERY [RVSON]G[RVSOFF]OOD [RVSON]B[RV
SOFF]OY [RVSON]D[RVSOFF]ESERVES [RVSON]F
[RVSOFF]UDGE"                            KO
•1260 GOSUB4300                          FM
•1265 PRINT"[4" "]BASS CLEF SPACES, THE S
ENTENCE:"                                LD
•1270 PRINTTAB(11)"[DOWN][RVSON][c 7]A[RV
SOFF]LL [RVSON]C[RVSOFF]OWS [RVSON]E[RVS
OFF]AT [RVSON]G[RVSOFF]RASS"             PG
•1280 GOSUB4300                          FM
•1285 PRINT"[5" "]BASS CLEF LINES, THE SE
NTENCE:"                                 PM
•1290 PRINT"[DOWN][6" "][RVSON][c 7]G[RVS
OFF]REAT [RVSON]B[RVSOFF]IG [RVSON]D[RVS
OFF]OGS [RVSON]F[RVSOFF]IGHT [RVSON]A[RV
SOFF]NIMALS"                             MD
•1295 GOSUB4300                          FM
•1300 PRINT"[4" "]THE NOTE C IN THE MIDDL
E OF THE"                                PL
•1305 PRINT"[DOWN][3" "]PIANO KEYBOARD IS
 CALLED [RVSON][WHITE]MIDDLE[RVSOFF] [RV
SON]C[RVSOFF].                           CI
•1310 POKE214,20:PRINT:PRINTTAB(15)"[c 7]
[s C][UP][s C][UP][s C][UP][s C][UP][s C
][UP][RIGHT][UP][c 6][s C][UP][s C][UP][
s C][UP][s C][UP][s C]"                  CO
•1315 POKEV+21,163                       MM
•1320 POKE214,5:PRINT:PRINT"[UP][RVSON][c
 7][s Y] [s -][SS][s -][SS][s -][SS][s -
][SS][s -][SS][s -][SS][s -][SS][s -][SS
][s -][c 6][SS][s -][SS][s -][SS][s -] [
s -][SS][s -][SS][s -][SS][s -][SS][s -]
[SS][s -][SS][s -][SS]"                  HA
•1325 GOSUB4300                          FM
•1330 PRINT"[YELLOW]  ON THE GRAND STAFF,
 MIDDLE C IS ON A"                       FO
•1335 PRINT"[DOWN][3" "]SPECIAL LINE CALL
ED A [RVSON]LEDGER[RVSOFF] [RVSON]LINE[R
VSOFF]"                                  OI
•1340 GOSUB4300:POKEV+21,0:GOTO900       II
•2000 GOTO2000                           EP
•3000 REM READ JOYSTICK                  EA
•3010 FORT=1TO80:NEXT:JV=PEEK(56320):FR=J
VAND16:JV=15-(JVAND15):RETURN            JM
•4000 RETURN:REM HELP                    JK
•4200 REM RANGE SELECTION                CE
•4205 IFL1=0THENN$="[F1]"                IJ
•4210 POKE214,22:PRINT:IFN$<>"[F1]"THEN42
20                                       IF
•4215 L1=11:L2=11:PRINT"  [WHITE]F1-[RVSO
```

```
N][c 6]TREBLE[RVSOFF]  [WHITE]F3-[c 7]BA
SS  [WHITE]F5-[c 6]GRAND [c 7]STAFF":RET
URN                                           IB
•4220 IFN$<>"[F3]"THEN4230                    LL
•4225 L1=11:L2=1:PRINT"  [WHITE]F1-[c 6]T
REBLE[RVSOFF]  [WHITE]F3-[c 7][RVSON]BAS
S[RVSOFF]  [WHITE]F5-[c 6]GRAND [c 7]STA
FF":RETURN                                    LA
•4230 L1=21:L2=1:PRINT"  [WHITE]F1-[c 6]T
REBLE[RVSOFF]  [WHITE]F3-[c 7]BASS  [WHI
TE]F5-[c 6][RVSON]GRAND[RVSOFF] [c 7][RV
SON]STAFF[RVSOFF]":RETURN                     BL
•4250 REM CLEAR AND POSITION CURSOR           JP
•4255 POKE214,6:PRINT:FORX=1TO12:PRINT"[1
3" "]";:NEXT                                  LJ
•4260 POKE214,22:PRINT:PRINTTAB(7)"[26" "
]"                                            NA
•4270 POKE214,6:PRINT:PRINT"[DOWN]";:RETU
RN                                            LO
•4300 REM INSTRUCTION CONTINUE                LO
•4310 POKE214,22:PRINT:PRINTTAB(7)"[BLACK
]PRESS ANY KEY TO CONTINUE.[YELLOW]"          GE
•4320 GETA$:IFA$=""GOTO4320                   JJ
•4330 GOSUB4250:RETURN                        CD
•4500 REM CORRECT ANSWER                      LI
•4505 POKE54275,10:POKE54274,0:POKEV+21,P
EEK(V+21)OR64                                 LL
•4510 POKE54296,15:POKE54273,A(NI):POKE54
272,B(NI):POKE54277,15:POKE54278,0            PC
•4515 POKE54275,10:POKE54274,0:POKEV+21,P
EEK(V+21)OR64                                 LL
•4520 POKE54276,65:FORX=1TO750:NEXT:POKE5
4276,64:POKE54296,0                           MF
•4525 POKEV+21,PEEK(V+21)AND191:RETURN        PF
•4600 REM WRONG ANSWER                        CG
•4610 POKE54296,15:POKE54277,45:POKE54278
,165:POKE54276,65:POKE54273,1                 GK
•4615 POKE54272,195:POKE54275,5:POKE54274
,120                                          DJ
•4620 FORX=1TO300:NEXT:POKE54276,32:POKE5
4273,0:POKE54272,0:RETURN                     ON
•5000 REM TREBLE CLEF                         MB
•5010 DATA,12,,,10,,,10,,,10,,,12,,,24,,,
56,,,104,,,200,,1,136,,3,62,,6,75,0           BF
•5020 DATA12,137,128,12,200,192,12,8,192,
6,8,192,3,137,128,,255,,,8,,,40,,,56,0        CM
•5025 REM BASS CLEF                           AL
•5030 DATA31,,,96,195,,192,99,,240,48,,11
2,48,,,51,,,51,,,96,,,192,,1,128,,3,,,12      KL
•5040 DATA,,48,,,192,,,,,,,,,,,,,,,,,,,,
,,0                                           IM
•5045 REM 4/4                                 FH
•5050 DATA,1,192,,3,192,,6,192,,12,192,,3
1,224,,,192,,1,224,,,,0                       AO
•5060 DATA1,192,,3,192,,6,192,,12,192,,31
,224,,,192,,1,224,,,,0                        NN
•5070 DATA,,,,,,,,,,,,,0                      BG
•5075 REM ACE                                 IH
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
•5080 DATA,,,,,248,,,192,,,192,,,252,,,,,
120,,,204,,,192,,,192,,,192,,,204,0           FE
•5090 DATA,120,,,,,48,,,120,,,204,,,252,,
,204,,,204,,,204,,0                           KP
•5100 REM NOTE BODY                           DM
•5110 DATA,,120,,,252,,1,254,,1,254,,1,25
4,,,252,,,120,0                               KG
•5120 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,0                                    BH
•5125 REM LETTER C                            PO
•5130 DATA120,,,204,,,192,,,192,,,192,,,2
04,,,120,,0                                   HM
•5140 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,0                                   GO
•5145 REM CLOWN                               DC
•5150 DATA,16,,,84,,1,85,,5,85,64,62,170,
240,254,34,252,62,154,240,10,154,128          NK
•5160 DATA9,169,128,8,84,128,9,1,128,1,85
,,63,171,240,63,207,240,63,19,240,,16,,,
,                                             BB
•5170 DATA,,,,,,,,,,0                         KL
•5175 REM MIDDLE C                            MJ
•5180 DATA255,240,,255,240,,255,240,,255,
240,,255,240,,255,240,,255,240,,255,240       GC
•5185 DATA0                                   LH
•5190 DATA255,254,,255,254,,255,254,,255,
254,,255,254,,255,254,,255,254,,255,254       FK
•5195 DATA,,,,,,,,,,,,,,,0                    FL
•5200 REM EGB                                 IK
•5210 DATA,,,,,,,,248,,,204,,,204,,,248,,
,204,,,204,,,248,,,,,120,,,204,,,192          AB
•5215 DATA,,220,,,204,,,204,,,120,,,,,252
,,,192,,,192,,0                               IC
•5220 REM DFA                                 IL
•5225 DATA,,204,,,252,,,204,,,204,,,204,,
,,,252,,,192,,,192,,,248,,,192,,,192          JM
•5230 DATA,,192,,,,,240,,,216,,,204,,,204
,,,204,,,216,,,240,,0                         PL
•5235 REM ACE                                 IH
•5240 DATA,,,,,,,,,,3,240,,3,,,3,,,3,224,
,3,,,3,,,3,240,,,,1,224,,3,48,,3,,,3          DD
•5245 DATA,,3,,,3,48,,1,224,,,,,96,,,240,
,0                                            NN
•6000 REM NOTE VALUES                         NL
•6010 DATA12,143,14,24,15,210,16,195,18,2
09,21,31,22,96,25,30,28,49,31,165,33,135      NF
•6020 DATA37,162,42,62,44,193,50,60,56,99
,63,75,67,15,75,69,84,125,89,131              GN
```

TO THE TOP

FROM PAGE 55

```
•4 GOTO90                                     PP
```

```
•5 POKEB,32:POKEC,32:B=INT(RND(1)*10)+768
0:C=INT(RND(1)*10)+7691:GOTO34                NP
•6 BO=0:B=INT(RND(1)*14)*22+7680:C=B+198:
G=B+110:RETURN                                EJ
•7 PRINT"[CLEAR][5"[DOWN]"][YELLOW][RVSON
] MAGIC WINGS LEVEL!":FORT=1TO4:POKEW,2
41:FORTT=1TO200:NEXT:POKEW,0                  LN
•8 NEXT:I=I+1:RETURN                          LD
•9 PRINT"[CLEAR]":A=8175:POKEV+1,10:PH=2:
GOSUB6:U=U+5:FORT=1TOU+20:S=INT(RND(1)*4
52)+7680                                      JO
•10 POKES,3:POKES+LL,3:POKEX,200:POKEX,0:
NEXT                                          GG
•11 POKEA,2:POKEA+LL,5:IFPEEK(B+1)=3THENP
OKEW,200:B=B+44                               BN
•12 POKEB,5:POKEB+LL,4:D=PEEK(197):IFPEEK
(C+1)=3THENPOKEX,200:C=C+44                   HO
•13 IFPEEK(G+1)=3THENPOKEW,135:G=G+44         NI
•14 POKEG,5:POKEG+LL,4:POKEC,5:POKEC+LL,4
:IFD<>64THENPOKEA,32:IFD=20THENA=A-1:         NE
•15 IFA=CORA=BORA=GTHEN51                     PO
•16 BO=BO+1:IFD=44THENA=A+1                   KF
•17 POKEW,0:POKEX,0:IFD=42THENA=A-22          LO
•18 IFPEEK(A)=3ORPEEK(A)=5THEN52              PB
•19 POKEC,32:POKEB,32:POKEG,32:C=C+1:B=B+
1:G=G+1:IFBO=22THENGOSUB6                     DD
•20 IFA<7702THEN23                            GB
•21 GOTO11                                    PG
•23 GOTO75                                    PA
•24 RESTORE:FORPU=1TO47:READJ:NEXT:FORPU=
1TO15:READJ:POKEW,J:FORII=1TO100:NEXT:PO
KEX,J                                         KB
•25 FORII=1TO100:NEXT:NEXT:RETURN             EA
•29 PRINT"[CLEAR]":POKE36869,255:W=36876:
POKEV,15:POKEV+1,14:H=7680:A=8153:B=7680
:PH=1:C=7701                                  HB
•30 U=U+5:FORT=1TOU:S=INT(RND(1)*386)+774
6:POKES,3:POKES+LL,4:POKEX,200:POKEX,0:N
EXT                                           CG
•31 X=36875:IFI=1THENGOSUB24                  IB
•34 IFPEEK(C+22)=3THENPOKEX,135:POKEC,32:
C=C-2                                         FH
•35 D=PEEK(197):IFPEEK(B+22)=3THENPOKEW,1
35:POKEB,32:B=B+2                             OH
•36 POKEB,0:POKEB+22,1:POKEC,0:POKEC+22,1
:IFD<>64THENPOKEA,32                          GB
•37 IFD=20THENA=A-1                           KN
•38 IFD=44THENA=A+1                           JI
•39 IFD=42THENA=A-22                          NG
•40 E=PEEK(A):IFE<2ORE=3ORA=BORA=CORA=B+2
2ORA=C+22THEN51                               KA
•41 IFA<7702THEN59                            FK
•42 POKEA,2:POKEA+LL,5:POKEB,32:POKEC,32:
B=B+22:C=C+22:POKEW,.                         LD
•43 POKEX,.:IFB>8164THEN5                     MC
•44 GOTO34                                    PF
•46 PRINT"[CLEAR][RVSON]CONGRATULATIONS[4
"!"]":PRINT"[3"[DOWN]"][RVSON]YOU HAVE G
OTTEN TO[4" "]LEVEL 32[4"!"]":GOSUB24         BC
•47 PRINT"[DOWN][DOWN][RVSON]GET READY[4"
."]":LW=9:FORT=1TO1000:NEXT:GOTO29            DB
•51 IFLW=9THENONPHGOTO29,9,59,75              JN
•52 POKEW+1,220:POKEW,135:POKEA+LL,7:FORY
=1TO15:POKEV,Y:POKEA,42:FORZ=1TO5:NEXT:P
OKEA,214                                      NE
•53 POKEA,42:FORZ=1TO5:NEXT:POKEA,214:FOR
Z=1TO5:NEXT:NEXT                              LO
•54 FORY=15TO0STEP-1:POKEV,Y:POKEA,42:FOR
Z=1TO10:NEXT:POKEA,214:FORZ=1TO10:NEXT:N
EXT                                           MO
•55 POKEV,15:POKEW+1,0:POKEW,0:POKE36879,
24:PRINT"[CLEAR][RVSON][BLACK][6"[DOWN]"
] ZAPPED AT STAGE,";I                         JN
•56 IFI>32THEN46                              AE
•57 PRINT"[RVSON][4"[DOWN]"[10"[RIGHT]"]H
IT[DOWN][3"[LEFT]"]ANY[DOWN][3"[LEFT]"]K
EY"                                           OA
•58 POKE198,0:WAIT198,1:I=1:U=30:UU=2:GOT
O29                                           OP
•59 GOSUB7:POKEX,0:PRINT"[CLEAR]";:PH=3:F
ORT=1TO500:PRINT"[YELLOW] ";:NEXT             OI
•60 KD=135:A=8175:B=7812:C=7769:G=7726:UU
=UU+2:FORT=1TOUU:S=INT(RND(1)*22)+7680:P
OKES,4                                        PK
•61 POKES+308,4:POKES+308+LL,2:POKES+LL,2
:NEXT:F=7900                                  BO
•63 POKEA+LL,5:POKEA,2:POKEB,3:POKEC,3:PO
KEF,3:POKEG,3:POKEG+3,3:POKEW+1,KD:IFA<H
THEN73                                        IG
•66 D=PEEK(197):KD=KD+4:P=A:IFD=20THENA=A
-1                                            IM
•67 A=A-22:IFD=44THENA=A+1                    DO
•69 POKEP,32:POKEP+LL,7:IFPEEK(A)=3ORPEEK
(A)=4THEN51                                   IA
•72 POKEB-3,32:POKEC-3,32:POKEF-6,32:POKE
G-3,32:F=F+1:B=B+1:C=C+1:G=G+1:GOTO63         FI
•73 I=I+1:POKEW+1,0:GOTO9                     LL
•75 PH=4:KD=135:GOSUB7:PRINT"[CLEAR]":POK
EV+1,15:A=8175:FORT=1TOUU*2:S=INT(RND(1)
*386)+7680                                    AH
•76 SS=INT(RND(1)*6)+1:POKES,5:POKES+LL,S
S:POKEX,200:POKEX,0:NEXT                      CD
•78 POKEA,2:POKEA+LL,5:POKEW+1,KD:KD=KD+4
:D=PEEK(197)                                  LH
•79 P=A:IFD=20THENA=A-1                       DI
•80 A=A-22:IFD=44THENA=A+1                    DO
•81 POKEP,32:IFPEEK(A)=5THEN51                MC
•82 IFA<HTHEN84                               ON
•83 GOTO78                                    PN
•84 POKEW+1,0:I=I+1:GOTO29                    JH
•90 POKE52,28:POKE56,28:CLR:FORI=7168TO76
79:POKEI,PEEK(I+25600):NEXT:FORH=7168TO7
216                                           FD
•91 READJ:POKEH,J:NEXT:POKE36869,255:PRIN
T"[CLEAR][8"[DOWN]"[9"[RIGHT]"][BLACK]B[
DOWN][LEFT][RVSON][RED]TO[DOWN][LEFT][LE
```

```
FT][BLUE]THE[DOWN][3"[LEFT]"][GREEN]TOP!
"                                          KA
·92 PRINT"[DOWN][DOWN][PURPLE][RVSON][4"
 "]BY KEVIN DEWEY":PRINT"[RVSON][DOWN][DO
WN][8"[RIGHT]"][BLACK]HIT[DOWN][DOWN][3"
[LEFT]"]ANY[DOWN][DOWN][3"[LEFT]"]KEY"    CL
·93 V=36878:POKE198,0:WAIT198,1           KG
·94 U=30:UU=2:LL=30720:I=1:GOTO29         GK
·95 DATA24,60,126,255,219,219,255,102,255
,219,219,227,126,60,36,102,255,86,126,16
,56,124                                   HF
·96 DATA40,108,255,219,219,255,165,189,12
9,255,127,8,127,93,93,127,20,34,126,219,
126                                       NJ
·97 DATA24,255,153,60,102,0               PG
·98 DATA200,140,145,150,140,150,180,140,1
30,200,210,190,150,140,0                  FH
```

Tape/Disk Transfer

FROM PAGE 25 — C-64 VERSION

```
·100 FORI=53181TO53247:READA:POKEI,A:NEXT NL
·110 PRINT"[CLEAR][5" "][RVSON] TAPE TO D
 ISK TRANSFER PROGRAM "                   OL
·120 PRINT:PRINT"INSERT AND REWIND TAPE T
 O BE COPIED"                             AA
·130 PRINT:PRINT"INSERT DISK INTO DISK DR
 IVE"                                     FC
·140 PRINT:PRINT"DOES THE DISK NEED TO BE
 FORMATTED? Y/N"                          MK
·150 GETA$:IFA$=""THEN150                 HN
·160 IFA$><"Y"THENSYS53181                FP
·170 INPUT"NAME OF DISK";A$               JE
·180 INPUT"DISK ID";B$                    FL
·190 PRINT"[CLEAR][8"[DOWN]"]"            AP
·200 PRINT"OPEN15,8,15"                   GL
·210 PRINT"[DOWN][DOWN]P[s R]15,"CHR$(34)
"N0:"A$","B$CHR$(34)                      BP
·220 PRINT"[DOWN][DOWN]SYS53181":PRINT"[1
0"[UP]"]"                                 OI
·230 POKE631,13:POKE632,13:POKE633,13:POK
E198,3:END                                IL
·910 DATA169,1,162,1,160,1,32,186,255,169
,0,162,65,160,3,32,189,255,169,0          FP
·920 DATA32,213,255,169,8,162,8,160,255,3
2,186,255,169,20,162,65,160,3,32          LL
·930 DATA189,255,173,61,3,141,251,0,173,6
2,3,141,252,0,169,251,174,63,3,172        BO
·940 DATA64,3,32,216,255,76,189,207       KJ
```

VIC 20 VERSION

```
·100 FORI=828TO828+66:READA:POKEI,A:NEXT  JK
·110 PRINT"[CLEAR][5" "][RVSON] TAPE TO D
 ISK TRANSFER PROGRAM "                   AH
·120 PRINT:PRINT"INSERT AND REWIND TAPE T
 O BE COPIED"                             FC
·130 PRINT:PRINT"INSERT DISK INTO DISK DR
 IVE"                                     JA
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 61-63 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
·140 PRINT:PRINT"DOES THE DISK NEED TO BE
 FORMATTED? Y/N"                          NI
·150 GETA$:IFA$=""THEN150                 HN
·160 IFA$><"Y"THENSYS828                  MO
·170 INPUT"NAME OF DISK";A$               DF
·180 INPUT"DISK ID";B$                    CI
·190 PRINT"[CLEAR][8"[DOWN]"]"            AP
·200 PRINT"OPEN15,8,15"                   GL
·210 PRINT"[DOWN][DOWN]P[s R]15,"CHR$(34)
"N0:"A$","B$CHR$(34)                      BP
·220 PRINT"[DOWN][DOWN]SYS828":PRINT"[10"
[UP]"]"                                   ON
·230 POKE631,13:POKE632,13:POKE633,13:POK
E198,3:END                                IL
·910 DATA169,1,162,1,160,1,32,186,255,169
,0,162,65,160,3,32,189,255,169,0          FP
·920 DATA32,213,255,169,8,162,8,160,255,3
2,186,255,169,20,162,65,160,3,32          LL
·930 DATA189,255,173,61,3,141,251,0,173,6
2,3,141,252,0,169,251,174,63,3,172        BO
·940 DATA64,3,32,216,255,76,60,3          CF
```

MID PRINT

FROM PAGE 41

```
·100 POKE53281,1:PRINT"[CLEAR][GREEN][3"[
DOWN]"]"                                  DA
·102 GOSUB410                             CN
·103 PRINTTAB(4)"[CLEAR][10"[DOWN]"]ENTER
 ([3"Z"]) WHEN COMPOSITION FINISHED"      NP
·104 FORC=1TO3000:NEXT                    MB
·105 PRINTCHR$(147);TAB(8)"[12"[DOWN]"]UP
PER OR LOWER CASE? (U/L)"                 EF
·107 GETC$:IFC$=""THEN107                 GB
·108 IFC$="L"THENPOKE53272,23:GOTO115     PG
·109 IFC$="U"THENPOKE53272,21:GOTO115     KP
·110 IFC$<>"L"ORC$<>"U"THEN107            JA
·115 DIMA$(25):PRINT"[CLEAR]"             EI
·120 FORN=1TO23                           ME
·130 PRINT"ENTER LINE #"N:INPUTA$(N):IFA$
(N)="[3"Z"]"THEN145                       BC
·140 NEXTN                                NA
·145 PRINT"[CLEAR]"                       HH
·146 W=10-INT(N/2)                        FB
·147 FORZ=1TOW:PRINT"[DOWN]";:NEXTZ       CC
·150 FORX=1TO23                           MG
·160 L=LEN(A$(X)):S=INT(L/2):T=20-S       JK
·165 IFA$(X)="[3"Z"]"THEN185              ON
·170 PRINTTAB(T)A$(X)                     NJ
·180 NEXTX                                NK
·185 PRINT"[HOME][c 7][RVSON]  PRESS F1 T
O PRINT SCREEN-F3 TO RENEW  "             DG
```

```
·190 GETF$:IFF$=""THEN190                              ID
·200 IFF$="[F1]"THEN300                                MH
·210 IFF$="[F3]"THENRUN                                LE
·300 IFPEEK(53272)=23THENM=7                           JM
·305 OPEN1,4,M:IFAN<>1THEN310                          BD
·307 FORF=1TO37:PRINT#1:NEXTF                          KP
·310 FORX=1TO24                                        MF
·320 IFA$(X)="[3"Z"]"THENCLOSE1:GOTO360                KE
·330 L=LEN(A$(X)):S=INT(L/2):T=20-S                    JK
·335 IFAN=1THENPRINT#1,CHR$(15)TAB(T+40)A
 $(X):GOTO350                                          OO
·340 PRINT#1,CHR$(14)TAB(T)A$(X)                       KD
·350 NEXTX                                             NK
·360 PRINT"[HOME][c 7][RVSON][5" "]F1 TO
 REPRINT-F3 TO RERUN[8" "]"                            ID
·370 GETR$:IFR$=""THEN370                              HD
·380 IFR$="[F1]"THEN300                                LL
·390 IFR$="[F3]"THENRUN                                LA
·400 GOTO370                                           CG
·410 PRINT"[10"[DOWN]"]DO YOU WANT ANNOUN
 CEMENT PRINT FORMAT?":PRINTTAB(15)"[DOWN
 ](Y/N)"                                               KI
·420 GETAN$:IFAN$=""THEN420                            DA
·430 IFAN$="Y"THENAN=1:RETURN                          GL
·440 IFAN$="N"THENAN=0:RETURN                          HF
·450 GOTO420                                           CG
```

SCUTTLEBUTT

Continued from page 35

COMPUTE YOUR ROOTS

As we learned several issues ago when we solicited our readers for information on genealogy programs, a great many Commodore users are interested in using their computers to get to the roots of their family trees. A new bimonthly publication, *Micro-ROOTS,* devotes itself to just this topic. For users of all micros at all levels, the journal provides research techniques, user group activities, reviews of genealogical software, and more. Subscription price is $24.95 per year.

Micro-ROOTS, 14208 Weeping Willow Drive, Suite #32, Wheaton, MD 20906 (phone: 301-460-0754).

DISC MANAGEMENT

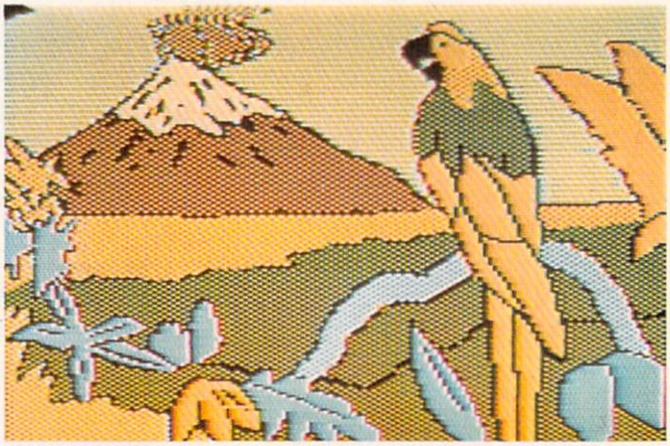
So you want to be a rock n' roll star? Forget singing on street corners, bub. This is 1984. K-tel, who know a little bit about the record business, reflect modern times in *It's Only Rock n' Roll.* The text and graphics adventure requires aspiring Elvises to balance their tour schedule, look out for lousy managers and lousier material, and keep the high life from ruining them even before they've made it. For the C-64.

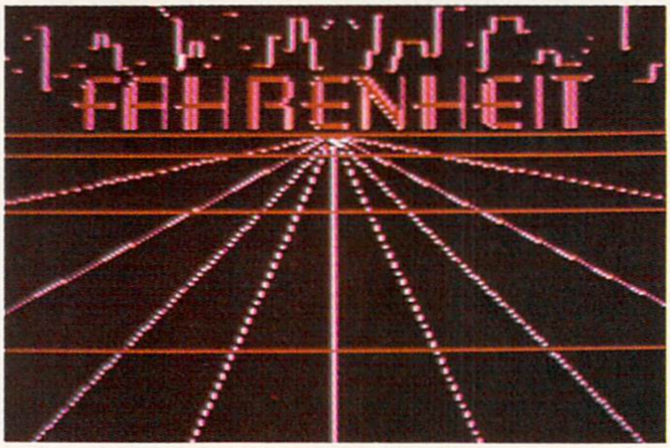
K-tel Software, Inc., 11311 K-tel Drive, Minnetonka, MN 55343 (phone: 612-932-4112).

GAMES BY SCI-FI PROS

We can't argue with the reasoning used by Spinnaker Software in the development of their Trillium series of text and graphics adventures: if you want a good science fiction game, hire a good science fiction writer. They hired some of the best to help translate their works to the computer screen, the authors in some cases writing the actual scripts for the games.

The first six Trillium releases will be *Rendezvous with Rama* by Arthur C. Clarke, *Amazon* by Michael Crichton, *Dragonworld* by Byron Preiss and Michael Reaves, *Starman Jones* by Robert Heinlein, *Fahrenheit 451* by Ray Bradbury, and *Shadowkeep,* from which a book has been adapted by Alan Dean Foster. Forthcoming games will be based on works by Phillip Jose Farmer, Roger Zelazny, Alfred Bester, and Harry Harrison.





Authors of the science fiction works adapted for the Trillium series got involved to varying degrees in producing the software. Michael Crichton wrote the script, worked with the graphic artist, and designed the software for* Amazon *(top). Below: Ray Bradbury's* Fahrenheit 451. *READER SERVICE NO. 86

In the role of the main character of each book, the player will be aided by assorted graphic clues that will make the games easier to play than most in the genre, but the various levels of complexity should still make for upwards of

40 hours of play per game. Some contain snippets of arcade-style play.

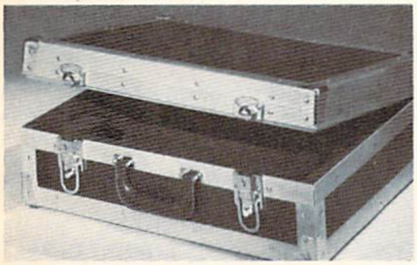
All on C-64 disk; $39.95 each.

Trillium Corp., division of Spinnaker Software, 1 Kendall Square, Cambridge, MA 02139 (phone: 617-494-1200).

PACKING HARDWARE

The Commodore SX-64 (reviewed next issue) provides computing mobility to those who need it—and can afford it. What about the 90% of us still reeling from the purchase of our home systems? Is there no alternative to remaining forever anchored at home?

Three Trav-L-Cases from the Computer Case Company are designed for the 64 or VIC. Each plywood case has nickel-plated steel edges and at least one inch of foam padding all around the inside. The cases are designed for a 64 or VIC plus dual drive ($129.00), single drive ($119.00), or datasette ($129.00).



One of three 64/VIC Trav-L-Cases.
READER SERVICE NO. 65

Computer Case Company, 5650 Indian Mound Ct., Columbus, OH 43213 (phone: 614-868-9464).

Portable Systems by MEI offers a soft briefcase that straps your computer in place and has numerous compartments for disks and other accessories.

Portable Systems by MEI, P.O. Box 15040, 1636 S. Second St., Fresno, CA 93702 (phone: 800-344-7422).

If you've got it and want to flaunt it to a ridiculous degree, you can spend $49.95 on a case for your diskettes. The Executive Disk Tote from Microcomputer Accessories, crafted of soft cordovan leather, will hold up to 12 disks tastefully.

Microcomputer Accessories, Inc., 5721 Buckingham Parkway, P.O. Box 3725, Culver City, CA 90231 (phone: 213-641-1800).

NEW MUPPET CAPERS

No strangers to software licensing, the Muppets will get still more exposure when CBS Software releases a series of programs based on the Fraggle Rock TV series "sometime in 1985." The entertainment/educational programs for the C-64 will be designed for children aged 7 to 12.

CBS Software, One Fawcett Place, Greenwich, CT 06836 (phone: 203-622-2500).



Teaches skills via familiar items.
READER SERVICE NO. 106

Koala's long-awaited Muppet Learning Keys, Kids' Computer Keyboard has at last come available for the 64. The child learns basic skills by pressing pictures of familiar items like a ruler, watercolor set, compass, eraser, and arithmetic board on a mylar surface above a circuit board. Characters like Kermit the Frog, Miss Piggy, Gonzo, and Fozzie Bear provide humorous instruction.

Price is $79.95.

Koala Technologies, 3100 Patrick Henry Drive, Santa Clara, CA 95052-8100 (phone: 408-986-8866).

FREE GAMES

Broderbund and WICO have extended the following offer to consumers who purchase any Broderbund game and the WICO Grip Handle Joystick after December 1: mail in the products' box tops, plus the coupon enclosed in the boxes, by February 15, 1985 and receive your choice of six Broderbund games: *David's Midnight Magic, Operation Whirlwind, Drol, Spare Change, Gumball,* and *A.E.*

Broderbund Software, 17 Paul Drive, San Rafael, CA 94903 (phone: 415-479-1170).

WICO Corporation, 6400 West Gross Point Road, Niles, IL 60648 (phone: 312-647-7500).

C-SIXTY-FORE

With the season over in most of the United States and Canada, *Golf Championship* can keep you in the swing of things. The Commodore 64 game lets you design your own course or play on one of the two supplied, with the option of selecting clubs, adding a hook or a slice to your swing, or playing with a foursome. Green fee: $29.95.

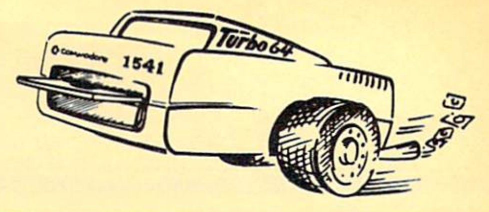
Adventure International, Box 3435, Longwood, FL 32750 (phone: 305-862-6917).

Of course, it's never to early to get set for next season. *Your Personal Golf Handicap* will determine yours, as well as compute your average score and putting average. C-64 tape or disk is $24.95.

F.A.R. Incorporated, 251 Green Street (no kidding), Somersworth, NH 03878 (phone: 603-692-3350).

TURBO 64

$19.95



Turbo 64 will turn your 1541 into a super fast and efficient disk drive. Loads programs five to eight times faster, works with 99% of your basic and machine language programs. The master disk allows you to put unlimited Turbo 64 boot copies on all your disks. This disk also includes:

Auto Run Boot Maker

Auto Run Boot Maker will load and auto run your basic or machine language programs.



$19.95

DMS-Errors 20,21,22,23,27 & 29
Format Single Tracks
Read Disk Errors

½ Track Reader-read and select ½ track.

½ Track Formatter-Format a disk with ½ tracks. This is where the next protection schemes are coming from.

Drive Mon-Disk Drive assembler/disassembler. For your 1541.

The Doc-Disk Doctor that reads code under errors.

Sync Maker-Place a sync mark on **any** track out to 41. Also used for protection.

Sync Reader-Check for Sync bits on **any** track out to 41.

Change Drive No.-Changes drive number (7-30).

Disk Logger-Finds starting track, sector; start and end addresses.

Disk Match-Compare any two diskettes. Byte for Byte.

New Wedge-Easier to use DOS wedge.

ID Check-Check ID's on **any** track.

Unscratch-Restore a scratched file.

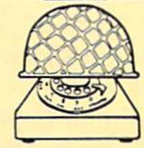
View RAM-Visual display of the free and used sectors on a diskette.

Read/Write Test-1541 performance test.

Repair a track-Repair a track with checksum errors. Reads code under errors and restores track.

Fast Format-Format a disk in just 10 seconds (with verify!).

WAR GAMES AUTODIALER



1-**Auto Dial** will automatically dial a set of numbers you choose.
2-**Review Numbers** will review numbers that were answered by a computer.
3-**Save Numbers** will save numbers where a computer answered.
4-**Hardcopy of Numbers** will print out list of numbers where a computer answered.
5-**LOAD Numbers** will load in numbers to continue where it left off.
6-**Continue** will pick up dialing where it was interrupted.

$29.95

GEMINI BIT COPIER



99.9% Effective!

- **3 Minute** copy program
- Copies Bit by Bit
- Eliminates worries of all Commodore DOS errors
- Very simple to use
- Half tracks
- 100% machine language
- Will not knock disk drive
- Copies quickly
- Writes errors automatically 20, 21, 22, 23, 27 & 29
- Errors are automatically transferred to new disk
- Copies identical syncs
- Supports use of two disk drives
- Unlocks disks to make your actual copies
- No need to worry about extra sectors
- This program covers all the latest protection schemes.

$29.95

SOFTWARE PROTECTION HANDBOOK

This book **"BLOWS THE LOCKS OFF"** protected DISKS, CARTRIDGES, and TAPES! Protection "secrets" are clearly explained along with essential information and procedures to follow for breaking protected software. An arsenal of protection breaking software is included with all listings, providing you with the tools needed! Programs include high speed error check/logging disk duplicator . . . disk picker . . . disk editor . . . Cartridge to disk/tape saver and several others for error handling and advanced disk breaking. The cartridge methods allow you to save and run cartridges from disk or tape! The tape duplicator has never been beaten! This manual is an invaluable reference aid including computer and disk maps, as well as useful tables and charts. (212 pages 11 programs)

C64 Book only .... $19.95 US
Book & Disk of all programs .... $29.95 US
Vic 20 book . . . Cart. & Tapes only .... $9.95 US

THIS MANUAL DOES NOT CONDONE PIRACY

*SHIPPING: $2.00

BULLETIN BOARD

Set up and operate your own bulletin board with a single 1541 disk drives. This one has all the features and you can customize it easily yourself.

1-RUN MEGASOFT BBS
2-CREATE MEGA FILES
3-ADD TO SYSOP'S CORNER
4-NEW SYSOP'S CORNER
5-READ MESSAGES
6-SCRATCH MESSAGE
7-CYCLE MESSAGES
8-READ SYSOP MESSAGES
9-WRITE OPENING MESSAGE
10-READ LOG
11-CYCLE LOG
12-READ DOWNLOAD FILE
13-SCRATCH DOWNLOAD FILE
14-ADD TO OTHER SYSTEMS
15-CREATE OTHER SYSTEMS

$39.95

Two Drive System: Includes everything listed above plus direct file transfer, dule passwords, for signon and updown load sections and many more options!

$59.95

Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders. Canada orders must be in U.S. Dollars. VISA — MASTER CARD — C.O.D.

All Programs are for C-64.

VISA MasterCard

MegaSoft Limited

P.O. Box 1080, Battle Ground, Washington 98604
Phone (206) 687-5205

BOOK REVIEW

MUSIC AND SOUND FOR THE COMMODORE 64
by Bill L. Behrendt (Prentice-Hall, 1984; $14.95 for book or disk, $29.95 for both). 185 pages.

If you want to understand music synthesis, *Music and Sound for the Commodore 64* is a good place to start. Behrendt, a musician who also loves computers, set out to explain music synthesis, computers, and music to anyone who wants to work with the Commodore 64's SID chip and lacks one of these three areas of knowledge.

In addition to tutorial information, half the book is devoted to program listings for utilities and example programs for making music and sound. The book/disk package is a steal for the programs alone. Since the book includes full source code and extensive annotation on each program, it's a must for anyone who seriously plans to program the SID.

Serious is the keynote here. Behrendt's technical discussions are very well written. The explanations are clear and simple, but the subject is complex and requires considerable study. This book makes it as easy as possible, but the casual programmer who just wants a few sound effects can find less demanding books.

The author's brief explanation of music synthesis is an excellent introduction to the process. His coverage of the physics of sound is a joy of simplicity. When he gets to the specifics of the C-64, he explains what the numbers in the SID registers really do. For example, he provides a table which correlates the attack and decay times in milliseconds with the values in the SID register.

The author strips down his explanations to essentials. Most discussions of programming music on the C-64 include material on bit-switching. Behrendt uses tables in which the user adds up decimal numbers. Even to someone who understands bit-switching, Behrendt's system is handier than dealing with binary notation.

Music and Sound for the Commodore 64 is worth buying for its tutorials. The programs that come with it are pure gravy. Consider a program for developing envelopes. Behrendt leads the reader through using SOUNDEP to imitate a brass, reed, and string instrument in turn.

The other major program in the book is a music editor. This editor does not include graphic representations of notes. However, anyone who knows enough to put notes on a staff can use Behrendt's code. The program permits instant playback, full editing, and a printer dump for the ending values.

Besides the major programs, the author has many small ones, including some for generating music by mathematical formulae. He offers sound effects like an explosion, a buzzer, and a bell. The appendix of ready-made sounds includes SID register values and enough analysis so a programmer can understand how to generate other sound effects like escaping steam or a snare drum.

Music and Sound for the Commodore 64 is a book for serious musicians and programmers. However, even people who do not intend to get deeply into music synthesis may enjoy the programs that come from the book and consider the book/disk combination a good software buy. I do not recommend buying the disk without the book, as these programs would lose much of their value without Behrendt's excellent documentation.

—Annette Hinshaw

Alice in Adventureland

Continued from page 40

need them all to find the rabbit.

At times, you will probably consider using the pit or looking glass as a crude means of getting around when you don't have that device's key. For instance: you find yourself on the sixth level without the elevator and looking glass keys. If you want to get to the first level quickly, you might chance the looking glass, for in all likelihood it will bring you closer to your destination. But, aside from the risk of losing a key or gold coin, you could also end up in a room that has only doors. If you do not have the appropriate key(s) you are trapped. Furthermore, you could even find yourself in a room with no openings and no doors!

Since you are not likely to finish the game in a day, let alone an hour, you can SAVE your position for another time. Press f1 to end and you will be asked if you wish to continue. If so, you will be instructed to press RECORD/PLAY. To resume the game, instruct the computer that you are playing a continuation and you will be asked to press PLAY. Once LOADed, it will skip the menu and set you up in your previous room along with any keys and gold coins that you had. □

SEE PROGRAM LISTING ON PAGE 74

BRINGING THE SCREEN TO LIFE

Continued from page 20

sets all the video back to normal, but BASIC doesn't. So BASIC is still looking for screen memory where you had it while your program was running. The simplest solution is to disable RUN/STOP-RESTORE while the program is running by POKEing 808,234. Then, to end the program, press SHIFT, COMMODORE, or CTRL. The program will then jump to line 200, which carefully puts everything in video memory back the way it was, reenables RUN/STOP-RESTORE, and ends the program. It's just a little added complexity, but if you don't do it, the only practical way to end the program is to turn the machine off and on again!

Next month we'll look a little more at the theory of game design, including a review of a fascinating book by one of this country's best game designers, Chris Crawford. In the meantime, have fun designing your own animated characters and putting them together into screen displays that will make your game a world that players will believe—and want to live in. □ ***SEE PROGRAM LISTING ON PAGE 72***

COMMODARES

Continued from page 54

also to Hugh Rountree (Perry, FL) for his letters on tape and all his solutions to many *Commodares.*

Glenn D. Elliot (Rutherford, NJ) submitted the following solution to *Problem #9-3: Common Letters* which typifies the majority of the solutions submitted by the readers listed above:

```
•10 REM PROBLEM #9-3:
•20 REM COMMON LETTERS
•30 REM SUBMITTED BY
•40 REM GLENN D. ELLIOT
•50 INPUT "WORDS";A$,B$
•60 FOR A=1 TO LEN(A$)
•70 FOR B=1 TO LEN(B$)
•80 IF MID$(A$,A,1)=MID$(B$,B,1) THEN X=X
 +1 : GOTO 100
•90 NEXT B
•100 NEXT A
•110 PRINT X"LETTERS OF "A$" ARE IN "B$
```

As usual there is simply not enough space to print everyone's solutions and ideas. Hopefully the reward of solving these problems and seeing your name in genuine print will keep you writing to this column (there are no monetary rewards; *Commodares* offers you fame, not fortune). If you feel that your version of a solution is unique in some way, tell me about it. Generally only the shorter, tidier programs will be listed, but this is a clearinghouse for all your suggestions, programming hints, tips and questions. Happy programming. □

SPRITE BASICS

Continued from page 52

tial, and that we specified the write operation in the secondary address.

If you run the program with these lines added, you will still see the numeric data in groups of three displayed on the screen. The information will be simultaneously stored in an external file. Now we need a program to bring the data from that file back into the computer when we need it. That brings us to *Sprite Loader.*

SPRITE LOADER

The *Sprite Loader* program on page 71 may be used independently, or you may incorporate it into any of your programs which use sprites. It reads a sequential file created by the *Sprite Maker* program and stores the data into the memory block that you specify.

If you have run the two previous programs, then the sprite data is already stored in memory at the block you specified, and it is stored on tape or disk. You could now turn your computer off and later run the *Sprite Loader* program to put the sprite data back into memory.

You could also run the *Sprite Loader* program

now to put the same or additional sprite data into different memory blocks. Notice again the difference between the disk and tape versions. This program simply reads the twenty-four sprite data values from a specified sequential file into a specified block of memory.

Once the sprite data is stored in memory, you may load any other BASIC programs into memory without affecting the sprite data (assuming your BASIC programs are not too large). The sprite data does not have to be reloaded every time a different program is loaded or run.

So far there is nothing very complicated about what we've done. We have transformed a screen image from DATA statements into numeric data. We have POKEd that data into memory, and we have created a sequential file of that data on tape or disk. We have also created a program to bring the data back into memory from the tape or disk.

But, you may say (and rightfully so), there is still no image of this sprite on the screen! Let's proceed now to the final phase. Let's see what it takes to display this sprite.

SPRITE CONTROLLER

Assuming the sprite data is already stored in memory, we must still do several things. We must tell the VIC-II:

1. in which memory block the sprite data is stored.
2. what the foreground color of the sprite is.
3. where on the screen we want the sprite displayed.
4. when to turn the sprite on or off.

The *Sprite Controller* program on page 71 allows us to do all of these things.

The program first defines a couple of useful functions, FNSB and FNRB. These functions are used to set and reset a specific bit of a given memory location. The memory address must be stored in the variable M. You must change the function definition if you wish to use some variable other than M. You may, however, use any variable or constant for the bit number N which is stored as the argument of the function. Using these functions will greatly simplify the manipulation of sprite data, as we will see.

The VIC-II chip contains forty-seven registers (numbered zero through forty-six), located at addresses 53248 through 53294. In line 35 the variable V is given the value 53248. To put a value X into the VIC-II register zero, we simply POKE V,X. To put the value Y into register 35, we may POKE V+35,Y.

The VIC-II controls eight sprites, numbered 0 through 7. The VIC-II automatically looks into RAM starting at address 2040 to find pointers to the memory blocks containing our sprite data. We must put a block number from 0 to 255 into location 2040 to tell the VIC-II where data for sprite 0 will be found. Similarly location 2041 stores the block number pointing to sprite 1 data, and location 2047 contains the block number indicating the starting memory location for sprite 7's data. In general, if the sprite number is SN, the pointer to its image data is in address 2040 + SN.

As we have previously discussed, if the VIC-II finds the number 192 in RAM location 2040, it then uses sixty-three consecutive bytes of data starting at memory location 12288 (64 times 192) to create the screen image for sprite 0. The subroutine at line 500 sets up the pointer for a specified sprite when the *Sprite Controller* program is executed.

Perhaps the most confusing part of the sprite dis-

cussions in the *Programmer's Reference Manual* deals with the horizontal location of the sprite on the screen. This program takes care of all the problems of putting the sprite at any horizontal screen position from 0 through 511. All you do is enter the desired X value, and the subroutine at line 600 does the rest.

The difficulty comes from the fact that it takes nine bits of data to represent the 512 possible horizontal screen positions. The first 256 are easy. One byte of data is all that is necessary. The remaining bit is called the most significant bit (msb). It is stored in a single byte with the msb's of the other seven sprites. Consequently we must get down to the bit-manipulation level in order to locate a sprite on the screen.

Address V+16 is the byte containing the msb's of all sprites' horizontal positions. Line 600 sets M to V+16. Then, depending upon the value of X chosen, the specific bit of address M is set or reset in line 620 or 610. If the X value of sprite SN is less than 256, then the msb should be reset (set to zero). If the sprite is at a horizontal position greater than 255, its msb must be set.

The VIC-II register at V+SN*2 stores the least significant byte of the X position for sprite number SN. That is, V+0 is for sprite 0, V+2 is for sprite 1, ..., V+4 is for sprite 7. The chosen value is POKEd at line 630.

The vertical position of the sprite on the screen is easier to handle, since it ranges from 0 to 255 and is represented in one byte. That byte is in the register located at V+SN*2+1 for sprite number SN. Thus V+1, V+3, ... , V+15 correspond to sprites 0 through 7. The selected vertical (Y) value is POKEd by the subroutine at line 700.

The value for the foreground color of sprite number SN is stored in the VIC-II register at address V+39+SN. The color values are listed in Appendix O of the C-64 *User's Guide* and in Appendix G of the *PRG.* (The labels at the top of page 161 in the *User's Guide* are improperly placed. The third, fourth, and fifth columns of the table list the DEC, HEX, and COLOR values which we use.)

The subroutine at line 800 puts the chosen color value into the proper VIC-II register.

Just as one byte was used to store the msb's of all sprites' horizontal positions, there is one byte which keeps track of which sprites are currently on and which are off. That byte is at address V+21. Each bit which is set (1) corresponds to a sprite that is currently on, and each bit which is reset (0) corresponds to a sprite that is turned off.

Once again the functions defined in this program make easy work of setting and resetting individual bits within a given byte. We will look at those functions in more detail next month. For now, a simple call to the subroutine at line 900 sets the bit of the selected sprite so that the sprite is on.

After the computer executes line 150 of the program, the sprite is displayed on the screen, assuming that you have specified X and Y values that are in the visible portion of the screen and that you have chosen a color for the sprite that contrasts with the current background color of your screen.

To turn the sprite off, line 180 calls the subroutine at line 1000. The appropriate bit in location V+21 is reset, and the sprite vanishes.

It is also possible to make a sprite disappear without turning it off by simply moving it out of the visible portion of the screen. We will see more about that next month, but you may get a head start by studying Figure 3-3 on page 140 of the *PRG.* The small rectangles in the figure correspond to basic sprites, and the larger rectangles refer to expanded sprites. You may also make a sprite disappear by changing its color to the background color of your screen.

The only thing left to do is to make the sprite live up to its name and start moving it around on the screen. Although in some graphics systems it is possible to "give the sprite a shove" and let hardware take care of its movement, motion is a software function on the Commodore 64. Our program must continuously update the position registers in order to move the sprite.

Fortunately we now have the tools to easily change the sprite's position. The subroutines at lines 600 and 700 make it very straightforward.

The little routine starting at line 2000 shows how to continuously move the sprite diagonally across the screen. Motion continues in one direction until the sprite hits a "wall" at which time it caroms to a different direction. The invisible wall is defined in lines 2030 through 2060. The variables DX and DY determine how the X and Y positions of the sprite change with each iteration of the program. If either X or Y becomes too large or too small, the sign of DX or DY is reversed, and the sprite rebounds.

You may experiment with the boundary sizes as well as the step sizes, DX and DY. You might also combine the subroutines at lines 600 and 700 into a single routine for speed. If you define two or more sprites, you may move them all around simultaneously (or nearly so). Investigate the differences between simply moving the sprite versus turning the sprite off, moving it, and turning it back on again. The trade-offs will always be between speed and smoothness of motion.

These subroutines are meant to be building blocks which you should incorporate into your own programs. Hopefully these examples have answered most of your questions about sprites. Next month we will fill in the few remaining details to give you complete control over these graphic elves. □

SEE PROGRAM LISTINGS ON PAGE 71

Installing a Device # Selector Switch on the 1541 Disk Drive

By Tom Benford

Photos...Liz Benford



Tools/parts needed: electric drill and bit, wire cutters/pliers, rosin-core solder, soldering iron, hookup wire, Phillips screwdriver, 1541 drive.

Commodore's 1541 Disk Drive has been called many things, not all of which are complimentary. But the bottom line is that it's a decent drive, albeit slow, and it's the most popular disk unit in use by VIC and C-64 owners. But then again, who ever said you can't make a good thing better?

If one 1541 is good, then it stands to reason that two 1541's should be better, right? This is particularly true if you're a "heavy" user–a person who uses his drives to control a BBS, manage a database, or frequently makes backup copies of disks. When using two or more 1541's together, the first thing you'll have to take care of is assigning device numbers to each of the drives. The first one is no problem; that's always device #8. But how do you assign the second one the device #9? The 1541 manual tells you two ways to do it–the software method, and the hardware method. Now I'm going to tell you how to do it by a *third* method–what I call the "switchable hardware" method. But first, let's examine the two methods given in the user's manual and see where they're lacking.

The software method lists a short program whereby the user opens up a channel for data transmission and reception for device #9, the second drive. Aside from being a pain in the neck to have to enter and save for future use, this method doesn't work well (or at all!) with many available disk utility programs, e.g. *The Clone Machine* or *Diskey.*

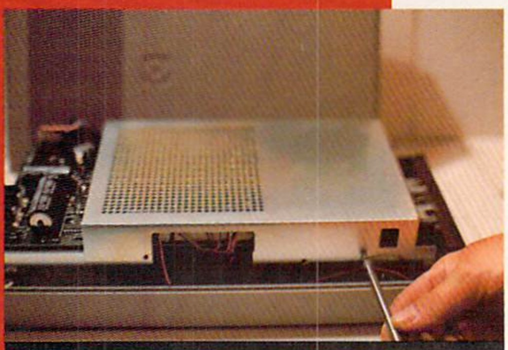
The hardware method involves partial disassembly of the drive to gain access to the device assignment jumpers. This method works well, but alas, once you've changed the jumpers the drive is *permanently* assigned to the new device #, which is not always desirable.

The alternative method I'm presenting here combines the best of both worlds: the non-permanence of the software method coupled with the convenience of the hardware method. This is accomplished by installing a toggle switch on the drive that will either connect the jumper (device #8) or disconnect it (device #9) simply by flipping the position of the switch. The whole modification will cost well under $5 and should take under ½ hour to complete. Before we get started on it, however, let me just say that while a great deal of technical prowess isn't needed, some basic soldering knowledge is. If you have any qualms about your ability in this area, it might be a good idea to take your drive, along with this article, to an electronics or TV repair shop and let them make the modification–if you elect to do this, it shouldn't raise the price of the modification to more than $15 total (any more than that and you're getting ripped off!). Please take notice that this modification will void your owner's warranty, if it's still in effect. Also note that, while this is a perfectly safe and effective modification, neither I nor *Ahoy!* can



Removing the four case cover screws on bottom of the 1541.

FIG. 1



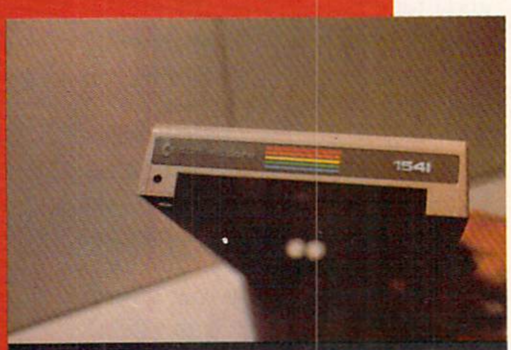
Removing two Phillips screws that hold on chassis cover.

FIG. 2



Drilling a 3/16" hole in the plastic top cover of drive case.

FIG. 3



The best location for the hole for device # selector switch.

FIG. 4

accept any responsibility for damage or repairs to your 1541. Now, let's get on with it.

Let's take inventory of the tools and materials you'll need to do this modification:

TOOLS

30-40 watt soldering iron with thin tip
Rosin-core electronics solder
Wire cutters/strippers
Phillips-head screwdriver
Electric drill with 5/16" bit
Fine-point scriber, X-acto knife or sharp, thin knife

MATERIALS

1 - SPST Submini Toggle Switch (Radio Shack #275-612 or equivalent), $1.59

18" - Thin stranded hook-up wire (24-gauge or finer) or wire-wrap wire or magnet wire (the thinner the better), approximately $.50

Once you've gotten your tools and materials together, you're ready to start. Find yourself a clean, well-lit work area and begin by turning the 1541 upside down on your work place. Using the Phillips screwdriver, remove the four screws on the bottom of the drive case, and put them in a safe place where they won't get lost (figure #1). Turn the drive right side up, facing you, and lift off the top cover. You'll see a metal cover with two small Phillips screws on the left side holding it on. Remove these screws also and store them safely (figure #2). Gently lift off this metal cover, and *voilà,* now you see the heart and soul of the disk drive. Now drill a 3/16" hole in the plastic top cover of the drive case, about ¾" down from the Commodore logo on the left side (figures #3 and #4), and set it aside for the moment.

The 1541 manual erroneously tells us that the device assignment jumpers will be found toward the middle of the board along the left edge—this is wrong! They will be found, instead, towards the front of the circuit board in the middle (see figure #5). Once you've located the jumpers, you'll notice that there are two of them; the one closer to the back of the drive controls assignments for device #'s 10/11, and the one closest to the *front* of the drive controls #'s 8/9. This is the one we're interested in. (A second toggle can be installed to control the rear jumper as well, but that's beyond the scope of this article.) If you look closely at the jumpers, you'll see that they're merely two silver semicircles connected by a thin foil bridge. It is by breaking this bridge that the device assignment number is changed. Obviously, once the bridge is broken, only by reconnecting it can the device number of changed back to its original status. That's where the switch comes in. By "closing" the circuit through the switch the drive is device #8. By "opening" or breaking the circuit with the switch, the drive is device #9. Simple enough logic, right? Okay, so now what you want to do is break the thin foil bridge of the jumper closest to the front of the drive. Use a fine scribe or an X-acto knife to do this, but be careful and work *gently*—it doesn't take much pressure to score through this foil bridge, and all that's necessary is to separate the two semicircles. Be careful not to damage any of the sur-

rounding components.

Now we're ready to solder. Cut your hookup wire into two 9" lengths, and strip about ¼" of insulation from all four ends. I strongly suggest "tinning" the wire ends with solder, since you'll be working on an extremely small area and flowing solder onto the jumpers will inevitably cause the jumpers to rebridge. Tinning the wire makes it much simpler to touch the iron to the wire on the jumper and create a bond. Solder one end of one wire to the left jumper half and one end of the other wire onto the right jumper half. Inspect your work to make sure that solder hasn't rebridged the gap that you separated with your knife. Once this is completed satisfactorily, you may then proceed to solder the remaining two wire ends to the toggle switch, each wire going to opposite terminals on the switch. After completing this, it's a good idea to try the drive out, just to make sure that everything's working as it should. Be careful not to touch any parts of the circuit board and keep your fingers out of the way. Plug in the power cord and the serial connector from your VIC or C-64, turn on the drive and the computer and try to load a disk-based program in the usual manner. If the computer gives you a "device not present" error, try flipping the switch and loading again. If everything works okay, then you're all set. Try to load the program using both the device #8 and device #9 calls, depending on the position of the switch. It should respond to both numbers, depending on how the switch is positioned. If it's not working in this manner, in all probability you've allowed some solder to reconnect the bridge between the jumpers, so you'll have to clean this up before proceeding.

Unplug the drive cables again, and route the wires across the printed circuit board (facing the front of the drive) to the left. Reposition the metal cover and install the two retaining screws. Turn the top cover of the drive over and place it next to the drive on the left hand side. Insert the switch into the hold, place the washer and nut on the switch, and tighten securely. Now gently turn the top half over, place it on the bottom half, invert the whole unit, and replace and tighten the four screws that hold the two halves together. That's it!

To be on the safe side, reconnect the power and serial cables, turn on the computer and the drive, and boot a program again, trying both the device #8 and device #9 calls. It's a good idea to label the switch positions so that you'll know which position indicates #8 and which indicates #9.

In most instances, when you're using two drives you'll find that it's best to turn the power on to the computer first, drive #8 second, and finally #9 third. For some reason the VIC and C-64 sometimes have a hard time telling how many peripherals are hooked up to it, and this method clears up the machine's "confusion." In the event that you're only going to be using one drive, simply leave the second drive turned off, and it will only acknowledge device #8. But at least now you have the flexibility of choosing your device # by the simple flick of a switch! □

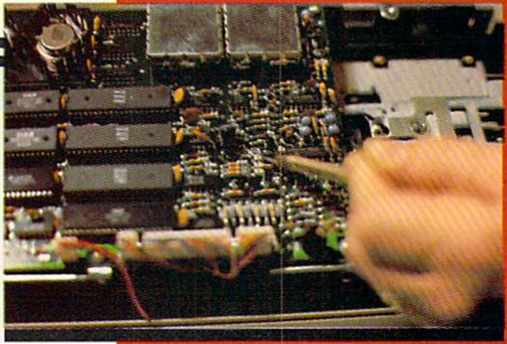


FIG. 5 *Location of device # jumper on current models of the 1541.*

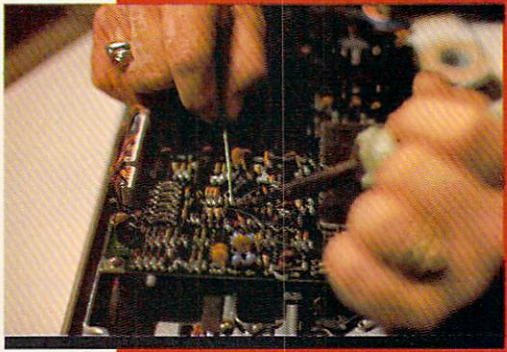


FIG. 6 *Soldering wires to device # jumper. Work carefully here!*

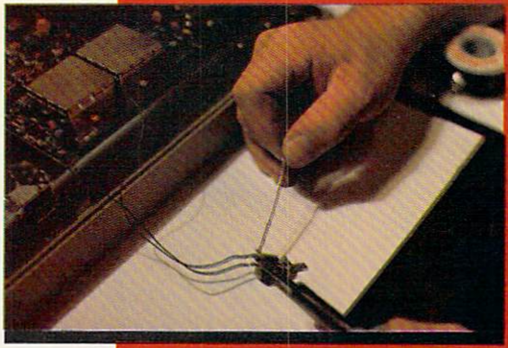


FIG. 7 *Soldering ends of hookup wire onto two switch terminals.*



FIG. 8 *The completed solder connections to jumper and switch.*

Put the World on Hold!



Cast off your cares and come sailing in the exotic Caribbean.

Not a dress-for-dinner floating hotel...but "barefoot" sailing & beachcombing for those with adventure in their souls. Lend a hand...or feet on the rail. Six exciting days from $425. Write for your free Great Adventure Book.



Windjammer 'Barefoot' Cruises
Box 120, Dept AD
Miami Beach, FL 33119-0120
or call TOLL FREE (800) 327-2600
in FL (800) 432-3364

Name ________________
Address ________________
City ________________
State/Zip ________________

Windjammer 'Barefoot' Cruises
Post Office Box 120, Miami Beach, Florida 33139

REVIEWS

Continued from page 49

transaction and the account you want to update. Then, depending on the type of account you've chosen, you will be prompted for pertinent information regarding the transaction. For example, if you've written a check you'll be asked to enter the date, check number, expenditure category (from the initiation list), payee, and amount.

Up to six deposits can also be registered for any given date, and a maximum of 150 transactions can be entered for each month. No matter what order you enter transactions in, they are always sorted by date.

Through the listing program, you can view or print out all expenses for a particular month, or data from a specific account. One of the most useful features of *Your Personal Accountant*, however, is the search routine program. This allows you to search for transactions by date, check number, payee, or category, and will even total the expenditures found on a search.

Your Personal Accountant probably won't save you any time. After all, if you've got to boot a finance program every time you write a check or make a deposit, you'll be spending a lot of time listening to your disk drive whirr. On the other hand, if you'd like some hard copy to show exactly where the paycheck goes every week, then *Your Personal Accountant* is worth waiting for.

SCRIPTIMUS

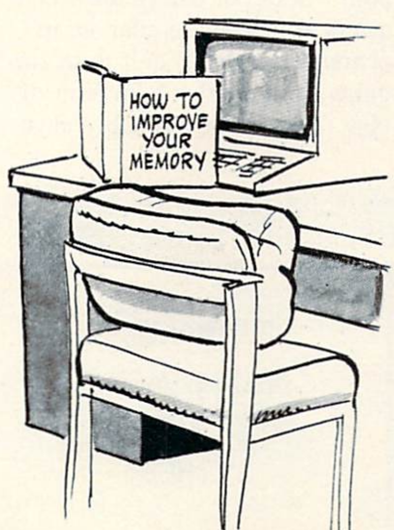
If you really need a good word processor you should pass up *Scriptimus* and buy one. In fact, you're better off typing everything five times on an antique manual typewriter than getting involved with this poorly written excuse for a WP.

The features that are included are complicated to employ and simply formatting your text for word wrap, right justification, and normal paging could take close to an hour. From the numerous warnings and descriptions of unusual circumstances in the instructions, I would guess that errors occur easily and, in fact, when I tried the Search and Replace function the program locked up and finally returned me to a completely blank screen. This one's not worth the disk it's saved on.

YOUR FILING CABINET

This final *Master Key* program lets you create a database of up to 200 records with up to fifteen fields each. A field may contain up to thirty-eight characters including both letters and numbers. Because the database is limited to only 200 records, the program is better suited to cataloging albums or recipes than it is to keeping business records.

To set up your DB, you must first decide how many fields you want for each record. The fields are then labeled and a maximum length is selected for each.



Once you've established the framework for your database you can easily enter, correct, and update your records. Outdated records can be deleted and the rest of the files "krunched" to eliminate empty records.

Records on file can be searched by any field and sorted either numerically or alphabetically by the contents of up to three fields. You can even sift by threshold by instructing the program to isolate all records with fields greater than or less than a certain amount. For instance, if one of the fields in your recipe file is "cooking time" and you need to whip up something in a hurry, you can sift for all recipes with cooking times of under one hour. Finally, the program lets you produce hard copy printouts of data lists. This is the best program on the *Master Key* disk.

CONCLUSIONS

It wouldn't be too difficult to find individual programs that greatly outshine those supplied on the *Master Key* disk. Whether you could assemble a similar software package for less than $129.00 (the retail price of *Master Key*) is doubtful. If you insist on having all five programs on one disk, then *Master Key*'s for you. If you don't mind paying a little more for some top-notch software, then you'd better shop around.

International Tri Micro, 1010 North Batavia, Suite G, Orange, CA 92667 (phone: 714-771-4038).

—Bob Guerra

LEARNING TO READ HEBREW

Davka Corporation
Commodore 64
Disk; $50 (with voice tape $75)

With this package, the Davka Corporation has succeeded in giv-

ing an old subject a modern twist. Owners of the Commodore 64 now have the option of learning to read the Hebrew language with the aid of their computer. This introductory tutorial is based on the traditional techniques which have been used for generations to successfully teach the fundamentals of reading Hebrew at the grade school level. It is designed to introduce the student with little or no previous experience to use the characters of the Hebrew alphabet.

The Hebrew alphabet uses a unique set of 24 letters which are read from right to left. A smaller set of associated symbols or vowels assign specific sounds to the various letters. Using a carefully structured series of fourteen lessons, the student is introduced to the various letters of the Hebrew alphabet. The program's pedagogy is excellent with continuous reinforcement of new concepts. Each lesson includes a series of quizzes which emphasize current material and continuously review previous work. The overall effect allows a novice to progress at a greater rate than initially expected.

The program is menu-driven. Considerable attention has been given to user-friendliness. The program is nearly self-explanatory, requiring a minimum of documentation. Good use was made of the Commodore 64's custom character capabilities to form the Hebrew alphabet. Transliterations of the more unusual Hebrew sounds was very well done, most notable being the "ch" sound (as in "Bach").

An optional sound tape is available for use in conjunction with the program. We cannot report on it as we did not receive a copy with our disk. The obvious value of auditory reinforcement with a program of this sort makes the tape a virtual necessity. We do feel it should be included with every package.

Supplied is a brief user instruction sheet. The program start-up sequence contains a significant error which is followed could damage the program disk. The user is instructed to insert the program disk into the drive before turning on the equipment. This procedure, while standard for the Apple, could be disastrous with the Commodore 64. The program disk should be inserted after all other associated equipment is turned on.

We did discover a few minor bugs. The program is long, filling both sides of a disk. Lesson eight on side one of the disk did not properly transfer control to lesson nine on side two. The disk had to be flipped and rebooted at that point. Some of the Hebrew characters which are similar in appearance were difficult to distinguish on the color television display. This would probably clear up with the Commodore color monitor. Some of the review quizzes required a letter key in response. This caused some hand-eye-brain coordination problems, especially when the letter selection was different than the letter response required. The program would do better if it stuck to all numerical responses for these parts of the quiz.

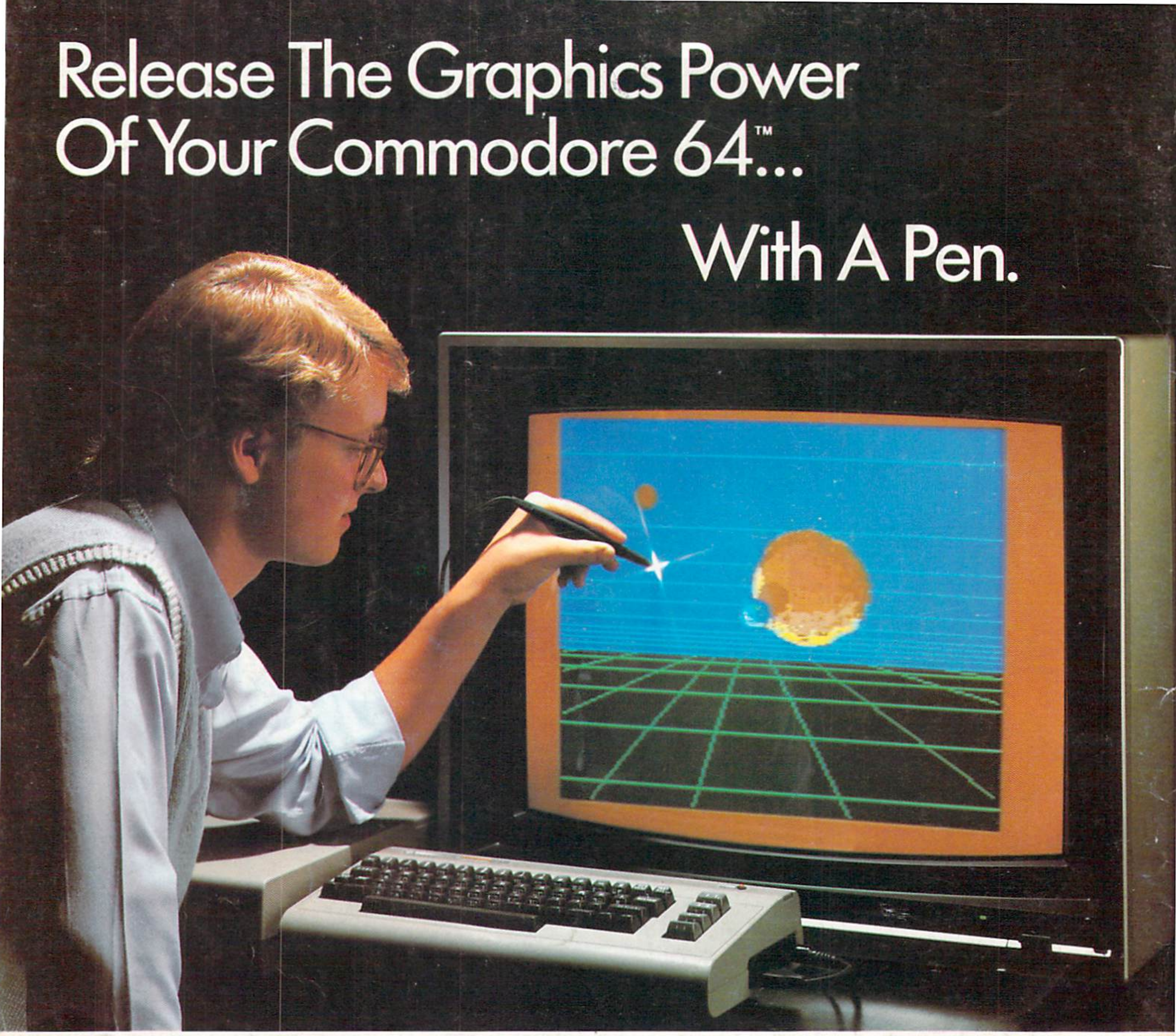
Finally, we would have liked to see some additional documentation on the subject. A bibliography of additional reference material is desirable. The Hebrew alphabet has several printed and written forms as well. A chart comparing the various type styles as well as the handwritten forms with character names would be a useful adjunct.

Learning to Read Hebrew is suitable for all English literates from the age of 10 to adult. It is directed toward anyone interested in learning to read Hebrew with little or no previous contact with the language.

Davka is actually a Jewish word which, loosely translated, might mean "for all this has been leading up to, the outcome is most certainly the result you should expect." As to the difference between Jewish and Hebrew, well, that is another story.

Davka Corporation, 845 North Michigan Avenue, Suite 843, Chicago, IL 60611.

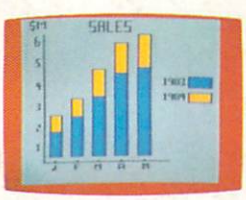
—Morton Kevelson



Now there's a sophisticated graphics system for your C-64—the Koala Light Pen System™ from Koala Technologies. It's easy. You work directly on your computer screen to draw, design, chart graphs. The Koala Light Pen System is all you need.

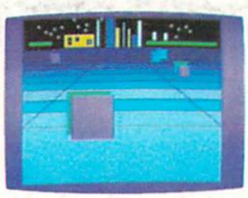
A State-Of-The-Art Light Pen

We've packed all the electronics in the streamlined Koala Pen, using the most advanced miniaturization techniques. This means clearer, smoother, more precise graphics—and, unlike other C-64 graphics products, you can use the full range of the C-64's colors. It also has an easy-to-use tip-switch.



A Complete Graphics Package

Koala, well known for our graphics systems, has created the premier C-64 graphics package. It contains a complete array of graphics capabilities. In addition, you can dip into a palette of 16 colors—and over 3,000 textured patterns. Point to "mirror" and a kaleidoscope of colorful images bursts onto the screen. Zoom in for 320 x 200 resolution. "Rubber band" lines and shapes.



Explore the full graphics power of your C-64 with the Koala Light Pen System. There is no other system like it.

The Koala Light Pen System™ and Graphics 64 Software.™



Koala Technologies Corporation
800-KOA-BEAR

Reader Service No. 97

K5006–0884